PEMBACA HURUF BRAILLE DENGAN MIKROKONTROLER MC68HC11

OLEH :

NEMUEL DANIEL PAH

NRP. 2882201112

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA

1994

PEMBACA HURUF BRAILLE DENGAN MIKROKONTROLER MC68HC11

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik Elektro
Pada
Bidang Studi Elektronika
Jurusan Teknik Elektro
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
S u r a b a y a

Mengetahui / Menyetujui
Dosen Pembimbing,

Ir. NAWANTOWIBOWO
NIP. 130 368 612

S U R A B A Y A
Pebruari, 1994

ABSTRAK

Komunikasi merupakan kebutuhan yang penting bagi semua umat manusia, tak terkecuali bagi mereka yang menyandang cacat tubuh. Bagi kaum tuna netra salah satu alat komunikasi adalah huruf Braille, yang berbentuk relief dari kombinasi enam titik. Untuk berkomunikasi dengan orang yang tidak menyandang cacat tuna netra diperlukan proses penerjemahan naskah Braille ke naskah latin. Pada Tugas Akhir ini dirancang dan dibuat alat yang dapat mengubah tulisan dalam naskah huruf Braille menjadi huruf latin yang dikirim ke komputer sehingga dapat dicetak, disimpan, atau diolah lagi sesuai kebutuhan. Pembaca relief huruf braille adalah berupa saklar sentuh yang mengubah relief menjadi besaran listrik. Sebagai pemroses utama digunakan mikrokontroler MC68HC11, yang dioperasikan pada mode expanded multiplexed yaitu dengan menggunakan memori eksternal. Alat ini diharapkan dapat membantu komunikasi antara kaum tuna netra dengan yang tidak tuna netra.

KATA PENGANTAR

Puji dan syukur kepada Tuhan Yesus Kristus karena berkat dan kasih karuniaNya, Tugas Akhir yang berjudul "PEMBACA HURUF BRAILLE DENGAN MIKROKONTROLER MC68HC11" ini dapat diselesaikan.

Tugas Akhir ini merupakan salah satu syarat dalam menyelesaikan pendidikan pada Jurusan Teknik Elektro Fakultas Teknologi Industri Institut Teknologi Sepuluh Nopember Surabaya.

Tugas Akhir ini dibuat berdasarkan teori-teori yang diperoleh pada bangku kuliah, serta berbagai buku literatur penunjangnya.

Dengan selesainya Tugas Akhir ini penyusun menyampaikan terima kasih yang sebesar-besarnya kepada :

1. Bapak DR. Ir. Moch. Solehudin, M.Eng,Sc., selaku Ketua Jurusan Teknik Elektro FTI-ITS,
2. Bapak Ir. Soetikno, selaku Ketua Bidang Studi Elektronika, Jurusan Teknik Elektro FTI-ITS,
3. Bapak Ir. Nawantowibowo, selaku dosen pembimbing,
4. Ibu DR. Ir. Handayani Tjandrasa, selaku dosen wali,
5. Seluruh staf dosen Bidang Studi Elektronika Teknik Elektro FTI-ITS,
6. Bapak Drs. Djoko Sayono, staf pengajar di Yayasan Pendidikan Anak Buta (YPAB) Tegalsari Surabaya,
7. Ayah dan Ibu, serta keluarga di rumah,

8. Sdri. Vivi Anggraini, S.Kom,
9. Seluruh staf dan karyawan di Jurusan Teknik Elektro FTI-ITS,
10. Teman-teman di Bidang Studi Elektronika Jurusan Teknik Elektro FTI-ITS,

atas segala bantuan, dorongan, dan bimbingan yang telah diberikan.

Penulis berharap kiranya Tugas Akhir ini bermanfaat bagi para pembaca, khususnya teman-teman di Bidang Studi Elektronika Jurusan Teknik Elektro Fakultas Teknologi Industri Institut Teknologi Sepuluh Nopember Surabaya.

Surabaya, Pebruari 1994

Penyusun

DAFTAR ISI

LAMPIRAN

- LISTING PROGRAM
- USULAN TUGAS AKHIR
- RIWAYAT HIDUP

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

I.1 LATAR BELAKANG

Dewasa ini perkembangan ilmu pengetahuan dan teknologi sangat pesat. Perkembangan ilmu pengetahuan dan teknologi ini meliputi segala bidang ilmu serta menyangkut segala sisi kehidupan manusia. Kebutuhan akan teknologi dalam era globalisasi tidak hanya terasa di perkotaan atau negara yang maju melainkan juga di tempat yang terpencil. Indonesia sebagai negara yang sedang berkembang juga mengembangkan dan mempelajari teknologi dan ilmu pengetahuan demi meningkatkan kesejahteraan masyarakatnya.

Salah satu bidang teknologi yang berkembang dengan cepat adalah teknologi elektronika. Perkembangan teknologi elektronika telah menyentuh semua bidang kehidupan dan berkaitan dengan teknologi-teknologi lainnya. Kehadiran teknologi elektronika telah membuat kegiatan-kegiatan atau proses-proses yang dulunya dilakukan secara manual menjadi suatu proses yang otomatis, cepat, akurat, serta lebih efisien.

Perkembangan teknologi elektronika dewasa ini semakin pesat dengan dikembangkannya teknologi elektronika digital serta rangkaian terpadu. Pembuatan prosesor-prosesor baru yang semakin cepat, dengan kemampuan proses yang tinggi dan akurat serta dengan bentuk yang semakin kecil telah menghadirkan alat-alat canggih, kecil, tetapi dengan

kemampuan yang semakin besar.

Kemajuan teknologi, khususnya teknologi elektronika diharapkan dapat membantu mensejahterakan kehidupan umat manusia termasuk bagi saudara-saudara kita yang dikaruniai Tuhan tubuh yang tidak selengkap tubuh manusia yang normal, atau yang biasa kita sebut orang cacat. Karena keterbatasan tubuh ini maka mereka pada umumnya lebih lambat dalam mengikuti perkembangan teknologi walaupun sebenarnya mereka membutuhkan kehadiran teknologi canggih untuk menggantikan fungsi dari bagian tubuh yang tidak mereka punyai. Sebagai sesama umat manusia kita terpanggil untuk membantu mereka dengan mengembangkan teknologi yang membantu kaum cacat.

Salah satu bentuk cacat tubuh adalah tuna netra. Bagi kaum tuna netra, kehilangan penglihatan telah membuat mereka kehilangan informasi yang biasa diterima melalui mata. Salah satu diantaranya adalah dalam hal membaca tulisan tercetak. Untuk menjawab kebutuhan ini dikembangkan sejenis huruf timbul yang kita kenal dengan huruf Braille. Dengan sistem huruf Braille ini mereka dapat membaca tulisan dengan menggunakan jari-jari mereka serta mereka juga dapat menulis dengan menggunakan mesin ketik huruf Braille. Dengan kehadiran alat-alat elektronik proses ini dapat dilakukan dengan lebih efisien baik dalam mencetak dan membaca huruf Braille maupun dalam mengkomunikasikan huruf Braille dengan sistem huruf lainnya, karena tidak dapat dipungkiri bahwa kaum tuna netrapun dapat menghasilkan pemikiran serta tulisan-tulisan yang ingin dibaca oleh mereka yang normal.

I.2 PERMASALAHAN

Untuk dapat menerjemahkan tulisan dalam bentuk huruf Braille ke tulisan latin atau sebaliknya, dibutuhkan seorang yang tidak tuna netra tetapi yang juga menguasai huruf Braille maupun huruf latin. Orang seperti ini sangat sulit ditemukan sehingga sampai saat ini proses menerjemahkan huruf Braille ke huruf latin dan sebaliknya masih sulit dilakukan.

Untuk menjawab permasalahan di atas diperlukan suatu peralatan elektronik yang dapat mengubah huruf Braille ke dalam besaran listrik kemudian besaran listrik tersebut diartikan sesuai dengan tata aturan tulisan huruf Braille dengan proses digital sehingga dapat dicetak dalam bentuk huruf latin sesuai dengan tata tulis huruf latin.

I.3 PEMBATASAN MASALAH

Dalam Tugas Akhir ini direncanakan dan dibuat suatu peralatan untuk mengubah tulisan dalam bentuk huruf Braille ke dalam tulisan dalam bentuk huruf latin.

Input dari peralatan ini berupa naskah huruf Braille. Huruf-huruf Braille yang ada pada naskah itu akan diubah menjadi besaran listrik oleh transducer. Besaran listrik ini kemudian diterjemahkan menjadi data dalam bentuk kode ASCII. Selanjutnya data ini dapat dikomunikasikan secara serial dengan komputer untuk diproses sesuai dengan keinginan pemakai, misalnya dicetak ke alat pencetak (printer) atau disimpan ke media penyimpan.

Sebagai pengolah data dan pengatur kerja dari peralatan ini digunakan mikrokontroler MC68HC11 dengan mode operasi expanded-mode. Sedangkan untuk mengolah data di komputer dapat digunakan semua jenis teks editor sesuai dengan kebutuhan pemakai.

I.4 TUJUAN

Perencanaan dan pembuatan peralatan pembaca huruf Braille dengan mikrokontroler MC68HC11 ini bertujuan untuk mempermudah proses penerjemahan huruf Braille ke huruf latin sehingga tulisan-tulisan dalam bentuk huruf Braille dapat dicetak dalam bentuk huruf latin maupun dapat diproses ke bentuk lain dengan bantuan komputer.

Selain tujuan di atas, perencanaan dan pembuatan peralatan ini juga bertujuan untuk mempelajari mikrokontroler MC68HC11 yang diharapkan dapat digunakan baik dalam peralatan ini maupun dalam peralatan lain yang membutuhkannya.

I.5 METODOLOGI

Pengerjaan Tugas Akhir ini dilakukan dengan langkah-langkah sebagai berikut:

- Studi literatur mengenai huruf Braille, mikrokontroler MC68HC11, IBM PC, elektronika industri, mikroprosesor, komunikasi serial, pemrograman komputer serta rangkaian elektronika.
- Perancagan perangkat keras dan perangkat lunak dari

peralatan.

- Pembuatan perangkat keras dan perangkat lunak sesuai dengan perencanaan yang telah dibuat.
- Pengujian serta perbaikan dari peralatan yang dibuat, dan proses ini dilakukan berulang-ulang sampai mencapai hasil yang diinginkan.
- Penulisan buku Tugas Akhir.

I.6 SISTEMATIKA PEMBAHASAN

Dalam buku Tugas Akhir ini, pembahasan mengenai peralatan yang dibuat terbagi dalam 7 bab dengan sistematika seperti dijelaskan berikut ini.

Bab I merupakan Pendahuluan yang membahas latar belakang, tujuan serta permasalahan dan pembatasan masalah dari pembuatan peralatan ini. Dalam bab ini juga dibahas metodologi dan sistematika penulisan buku Tugas Akhir ini.

Bab II berisi penjelasan mengenai huruf Braille menyangkut bentuk, tata tulis beserta contoh dari penulisan huruf Braille. Bab ini juga berisi teori penunjang meliputi mikrokontroler MC68HC11, dasar sistem elektronik, sistem mekanik, serta dasar teori mengenai perangkat lunak yang digunakan yaitu assembler MC68HC11.

Bab III berisi perencanaan perangkat keras.

Bab IV berisi perencanaan perangkat lunak menyangkut diagram alur, algoritma, serta prosedur-prosedur utama dalam perangkat lunak yang digunakan.

Bab V berisi karakteristik dan batasan kemampuan dari

peralatan yang diperoleh dari pengujian peralatan, serta cara pengoperasian peralatan.

Bab VI merupakan bab penutup yang berisi kesimpulan yang diperoleh dari pembuatan Tugas Akhir ini, serta saran-saran untuk pengembangan dari Tugas Akhir ini.

I.7 RELEVANSI

Peralatan ini diharapkan dapat membantu menterjemahkan naskah-naskah dalam bentuk huruf Braille ke dalam huruf latin sehingga dapat diolah selanjutnya dengan komputer misalnya dicetak atau disimpan dalam media penyimpan. Dengan bantuan mesin cetak huruf Braille maka naskah-naskah dalam bentuk huruf Braille dapat diperbanyak.

BAB II

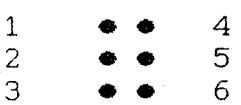
TEORI PENUNJANG

II.1 HURUF BRAILLE

Tulisan Braille diciptakan oleh seorang berkebangsaan Perancis yang bernama Louis Braille. Setiap karakter dalam tulisan Braille terdiri dari 6 (enam) buah titik relief. Setiap jenis kombinasi dari keenam titik relief tersebut mewakili satu karakter tertentu. Sistem tulisan Braille tersebut selesai dirancang secara lengkap pada tahun 1836, digunakan untuk keperluan bahasa, berhitung, dan musik. Jenis tulisan ini khusus ditujukan untuk kaum tuna netra, karena pembacaan tulisan ini dapat dilakukan dengan cara meraba.

Dalam suatu konggres di kota Paris pada tahun 1860 tulisan Braille diterima sebagai tulisan resmi bagi sekolah-sekolah tuna netra di seluruh Eropah Barat. Dari Eropah Barat tulisan Braille menyebar ke Amerika Serikat, Asia, Afrika, Australia, dan diperkenalkan di Indonesia pada tahun 1901 dengan berdirinya Blinden Institut di Bandung.

Seperti disebutkan di atas, tiap karakter dalam huruf Braille terdiri dari 6 buah titik relief. Untuk setiap titik dari keenam titik tersebut diberi nomor tetap 1 - 6, sesuai dengan posisinya. Penomoran titik-titik tersebut adalah sebagai berikut :



Dengan bantuan nomor-nomor tersebut, maka suatu karakter dapat dinyatakan sebagai suatu kombinasi dari nomor-nomor yang ada.

Tulisan Braille dibaca dari kiri ke kanan. Titik-titik yang digambarkan tebal menunjukkan bahwa titik-titik tersebut timbul. Berikut ini diberikan beberapa contoh karakter dan penamaannya menurut nomor titik-titik timbul :

titik 1

titik 1-2-3-5

titik 2-4-5-6

Pada tahun 1901 Dr. Westhoff mendirikan Blinden Institut di Bandung, sejak itu maka tulisan Braille mulai diperkenalkan di Indonesia. Sistem tulisan Braille yang dikenal saat itu mengikuti sistem yang berlaku di negeri Belanda.

Setelah Indonesia merdeka, pendidikan anak-anak tuna netra mengalami perkembangan yang lebih pesat dibandingkan dengan masa-masa sebelumnya. Sejalan dengan hal itu maka dirasakan perlu adanya keseragaman pemakaian tulisan Braille untuk berbagai istilah dan simbol-simbol untuk keperluan bahasa, matematika, ilmu pengetahuan alam, ilmu hayat, kimia, dan lain-lain. Untuk menjawab keperluan tersebut maka disusunlah pedoman tulisan Braille yang disesuaikan dengan

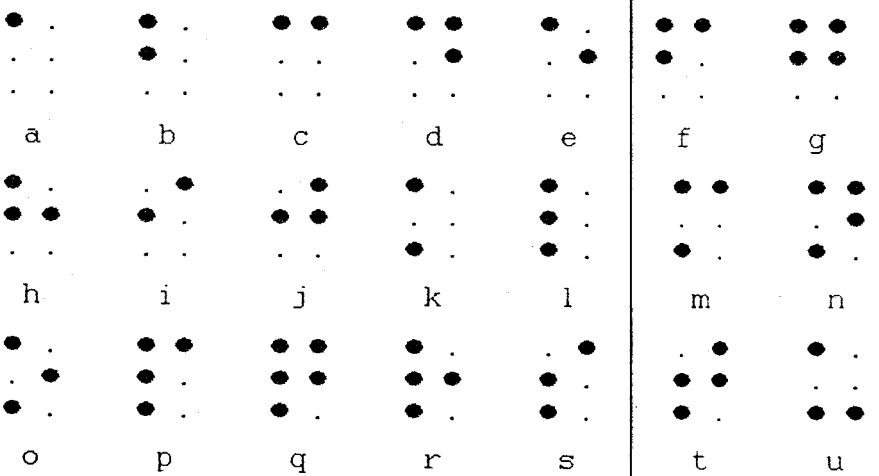
ejaan dalam bahasa Indonesia, sehingga dengan demikian maka terdapat pedoman tulisan Braille yang seragam untuk keperluan pendidikan di sekolah-sekolah tuna netra di Indonesia.

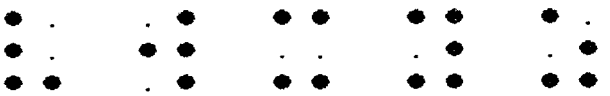
Pedoman tulisan Braille yang telah disusun beberapa kali mengalami penyempurnaan di sana sini, sesuai dengan perkembangan ejaan dalam bahasa Indonesia. Pada saat ini pedoman untuk tulisan Braille yang berlaku di Indonesia mengikuti buku "Pedoman Menulis Braille menurut Ejaan Baru Yang Disempurnakan". Dalam buku tersebut semua aturan untuk tulisan Braille sama dengan yang berlaku dalam buku pedoman menurut EYD tersebut.

II.1.1 Abjad

Abjad yang digunakan dalam bahasa Indonesia adalah abjad latin, yang berjumlah 26 (dua puluh enam), terdiri dari huruf a sampai z.

Dengan menggunakan tulisan Braille, huruf-huruf tersebut dinyatakan sebagai berikut :





v w x y z

II.1.2 Konsonan Rangkap dan Vokal Rangkap

Dalam bahasa Indonesia terdapat konsonan rangkap dan vokal rangkap yang menimbulkan satu bunyi bahasa. Konsonan dan vokal rangkap serta tulisan Braille yang mewakilinya adalah sebagai berikut :

Konsonan rangkap :

ng ny

Vokal rangkap :

ai au

II.1.3 Tanda Baca

1. Tanda titik

Pemakaian tanda titik :

a. Untuk mengakhiri kalimat yang bukan pertanyaan atau seruan.

b. Dipakai di belakang singkatan nama orang, gelar, jabatan, sapaan dan atau ungkapan.

c. Tidak digunakan dalam menuliskan bilangan besar untuk memisahkan angka ribuan, jutaan, dan sebagainya. Dalam menuliskan waktu tidak dipakai untuk memisahkan angka jam dari angka menitnya.

2. Tanda koma

,

Pemakaian tanda koma adalah sama dengan pemakaian pada Ejaan Yang Disempurnakan (EYD).

3. Tanda titik koma

;

Pemakaian tanda titik koma adalah sama dengan pemakaian pada Ejaan Yang Disempurnakan (EYD).

4. Tanda titik dua

:

Pemakaian tanda titik dua adalah sama dengan pemakaian pada Ejaan Yang Disempurnakan (EYD).

5. **Tanda seru**

!

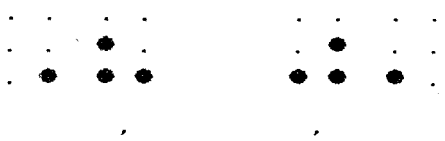
Pemakaian tanda seru adalah sama dengan pemakaian pada Ejaan Yang Disempurnakan (EYD).

6. **Tanda petik**

" "

Pemakaian tanda petik adalah sama dengan pemakaian pada Ejaan Yang Disempurnakan (EYD).

7. **Tanda petik tunggal**



' '

Pemakaian tanda petik tunggal adalah sama dengan pemakaian pada Ejaan Yang Disempurnakan (EYD).

8. **Tanda kurung**

( )

Pemakaian tanda kurung adalah sama dengan pemakaian pada Ejaan Yang Disempurnakan (EYD).

9. Tanda kurung siku

[ ]

Pemakaian tanda kurung siku adalah sama dengan pemakaian pada Ejaan Yang Disempurnakan (EYD).

10. Tanda ulang

2

Pemakaian tanda ulang :

a. Hanya dipakai pada tulisan singkat.

b. Dalam tulisan penuh, tanda ulang hanya digunakan untuk menulis kata ulang yang menyatakan satu pengertian; dan tidak untuk kata ulang yang menyatakan pengertian jamak.

11. Tanda hubung

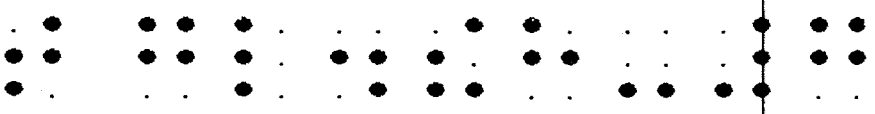
–

Pemakaian tanda hubung :

a. Untuk menghubungkan kata, bagian kata, yang terputus oleh pergantian baris.

b. Untuk menghubungkan kata ulang, misalnya : anak-anak, bermain-main, berlari-lari.

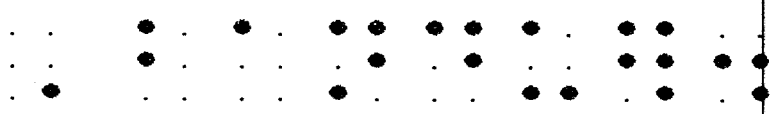
c. Untuk menghubungkan dua bilangan atau tanggal yang berarti sampai dengan, misalnya :



tgl. 5 - 7

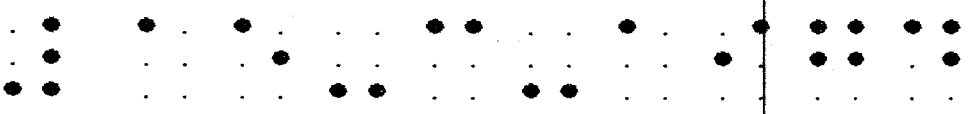
Untuk menghubungkan dua nama kota yang berarti ke atau sampai :

Jakarta -



Bandung

Untuk menghubungkan tanggal, bulan, tahun, yang ditulis dengan angka :



15 - 3 - 1974

12. Tanda pisah

—

Tanda pisah ditulis tanpa spasi dari huruf atau tanda yang mendahului atau mengikutinya.

Pemakaian tanda pisah :

a. Untuk membatasi penyisipan kata, kelompok kata atau anak kalimat yang memberikan penjelasan khusus.

b. Untuk menegaskan adanya oposisi atau penjelasan lainnya.

13. Tanda garis miring

Tanda garis miring ditulis tanpa spasi dari huruf atau tanda yang mendahului atau mengikutinya.

14. Tanda huruf besar

Pemakaian tanda huruf besar :

a. Tanda huruf besar ditulis rapat tanpa spasi dengan huruf yang dinyatakan sebagai huruf besar
b. Penggunaan huruf besar adalah sama dengan pada tulisan latin sesuai dengan EYD.
c. Bila beberapa kata huruf permulaannya ditulis dengan huruf besar, maka :
   1. Untuk satu sampai tiga kata, tiap kata didahului dengan tanda huruf besar.

      Contoh :

      Negara

      Republik

      Indonesia.

2. Untuk lebih dari empat kata, maka di depan kata yang pertama diberi dua tanda huruf besar, sedang di depan kata yang terakhir diberi sebuah tanda huruf besar.

   Contoh :

   Dari Sabang

   sampai Me-

   rauke.

3. Ketentuan di atas tidak berlaku untuk penulisan judul karangan, bab, atau fasal dan sebagainya.

   Untuk judul karangan, bab, atau fasal yang huruf permulaan tiap kata huruf besar, meskipun jumlah kata lebih dari tiga kata, tiap kata didahului dengan tanda huruf besar.

4. Menuliskan sebuah kata yang seluruh hurufnya huruf besar, di depan kata itu diberi tanda huruf besar rangkap.

   Contoh :

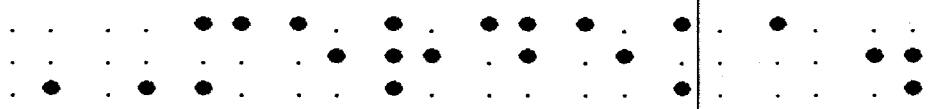
   MERDEKA

   Untuk menuliskan dua kata yang semua hurufnya ditulis dengan huruf besar, di depan tiap kata diberi tanda

huruf besar rangkap.

Contoh :

TETAP

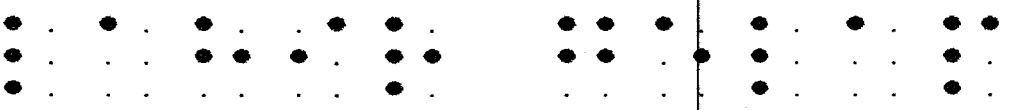


MERDEKA

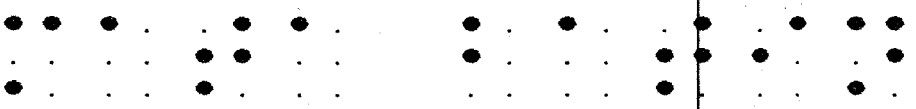
Untuk menuliskan tiga kata atau lebih yang semua hurufnya ditulis dengan huruf besar, di depan kata yang pertama diberi tiga tanda huruf besar sedang di depan kata yang terakhir diberi dua tanda huruf besar.

Contoh :

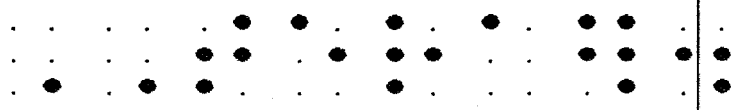
MATA



LAHIR GELAP



MATA BATIN

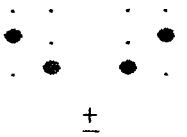


TERANG.

Dua buah ketentuan di atas tidak berlaku untuk menulis judul buku, karangan, bab, yang seluruhnya ditulis dengan huruf besar. Untuk hal-hal tersebut, di depan tiap kata diberi dua tanda huruf besar.

Dua kata yang dihubungkan dengan tanda penghubung dan masing-masing kata semuanya ditulis dengan huruf besar (kecuali kata ulang) masing-masing kata didahului dengan tanda huruf besar rangkap.

15. Tanda lebih kurang



±

Pemakaian tanda lebih kurang :

a. Tanda lebih kurang dipisahkan oleh satu spasi dari huruf atau tanda baca yang mendahului atau mengikutinya.
b. Tanda lebih kurang tidak dipisahkan oleh spasi dari angka atau tanda singkatan mata uang, ukuran yang mengikutinya.
c. Tanda lebih kurang dapat digantikan oleh singkatan, l.k.

16. Tanda bintang

*

Pemakaian tanda bintang :

a. Tanda bintang ditulis langsung tanpa spasi di belakang kata yang diterangkan, dan dipisahkan oleh satu spasi dari huruf atau tanda yang mengikutinya.
b. Bila dalam satu halaman terdapat lebih dari satu kata bertanda bintang, maka tiap tanda bintang langsung diikuti oleh angka penunjuk nomor urut.
c. Keterangan dari kata bertanda bintang ditulis pada bagian

bawah halaman dengan didahului oleh tanda bintang dan dipisahkan oleh satu spasi.

II.1.4 Peraturan Umum Penulisan Braille

Penulisan Alinea

Penulisan alinea dimulai pada petak ke tiga dari garis margin.

Penulisan nomor halaman

Penulisan nomor halaman di sebelah kanan atas dengan menggunakan tanda angka tanpa diikuti tanda titik.

Bila menggunakan tanda hubung pada nomor halaman, sebelum tanda hubung digunakan tanda angka dan setelah tanda hubung tidak lagi digunakan tanda angka.

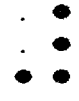
Penulisan judul

Penulisan judul bab, pasal, karangan dimulai pada petak keenam dari garis margin.

II.1.5 Matematika

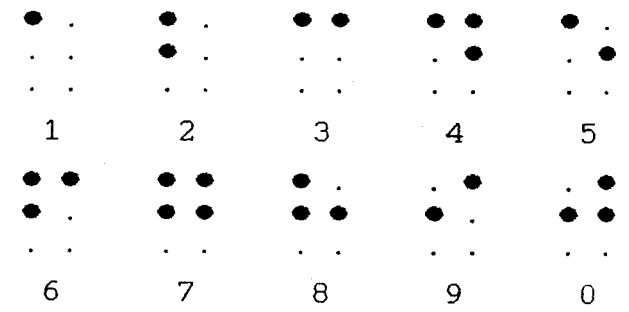
A. Bilangan / Angka

Dalam tulisan Braille, kesepuluh buah bilangan diwakili oleh sepuluh huruf pertama dalam abjad dengan ditambah tanda angka di depannya.

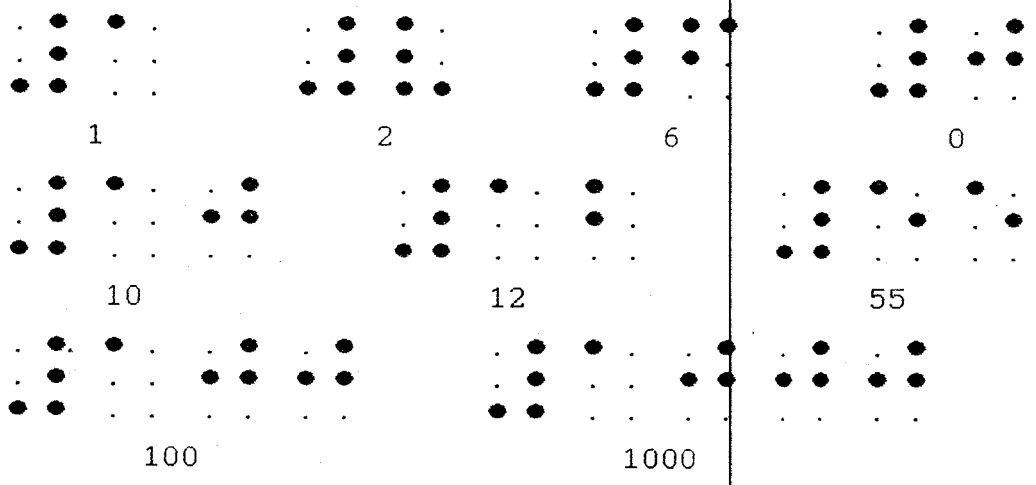
Tanda angka : 

yaitu terdiri dari titik 3, 4, 5, dan 6.

Sedangkan kesepuluh buah bilangan dinyatakan dalam tulisan Braille, sebagai berikut :



Berikut ini beberapa contoh penulisan angka dalam tulisan Braille :



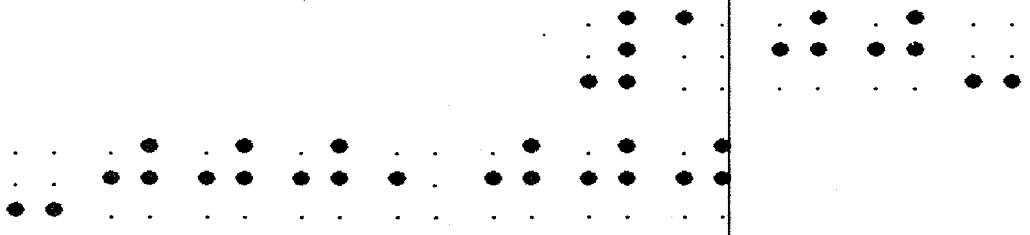
Dalam menuliskan bilangan 10.000 ke atas, digunakan aturan-aturan sebagai berikut :

1. Untuk memisahkan ribuan, jutaan, dan seterusnya, digunakan tanda :

Contoh :

2. Dalam menuliskan bilangan yang besar sebaiknya tidak deceraikan pada akhir baris. Tetapi jika bilangan terlalu besar sehingga terpaksa diceraikan pada akhir baris, maka pemisahan dilakukan dengan menggantikan titik pemisah dengan tanda penghubung dan pada permulaan baris diberi sebuah tanda penghubung lagi tanpa mempergunakan tanda angka lagi.

   Contoh :



   100.000.000

3. Beberapa bilangan yang dihubungkan dengan tanda-tanda penghubung, hanya diberi satu tanda angka saja, yaitu sebelum angka pertama.

   Contoh :

   17-8-1945

B. Tanda-tanda pengerjaan

Jenis - jenis tanda pengerjaan yang dimaksud adalah tanda tambah (+), tanda kurang (+), tanda kali (x), tanda bagi (:), dan tanda sama dengan. Dalam tulisan Braille tanda-tanda tersebut dinyatakan sebagai berikut :

Tanda tambah (+)

Tanda kurang (−)

Tanda kali (x)

Tanda bagi (:)

Tanda sama dengan (=)

II.2 SANDI MORSE

Selain kode dalam bentuk huruf Braille, peralatan yang dibuat juga menghasilkan bunyi dalam bentuk sandi morse. Sandi morse terdiri dari titik dan garis yang dikombinasi untuk mewakili huruf dan angka, dengan tidak membedakan huruf besar atau kecil. Sandi morse adalah sebagai berikut:

A : ._
B : −...
C : −.−.
D : −..
E : .
F : ..−.
G : −−.
H : ....
I : ..
J : .−−−
K : −.−
L : .−..
M : −−

N : −.
O : −−−
P : .−−.
Q : −−.−
R : .−.
S : ...
T : −
U : ..−
V : ...−
W : .−−
X : −..−
Y : −.−−
Z : −−..

1 : .−−−−
2 : ..−−−
3 : ...−−
4 : ....−
5 : .....
6 : −....
7 : −−...

8 : ——— ——— ——— · ·
9 : ——— ——— ——— ——— ·
10: ——— ——— ——— ——— ———

Tanda baca:

. : · ——— · ——— ——— · ———
, : ——— ——— · · ——— ———
? : · · ——— ——— · ·
– : ——— · · · · ———
: : ——— ——— ——— · · ·
_ : · · ——— ——— · ———
" : · ——— · · ——— ·
(: ——— · ——— ——— ·
) : ——— · ——— ——— · ———
= : ——— · · · ———
+ : · ——— · ——— ·
× : ——— · · ———

Aturan-aturan dalam penggunaan sandi morse adalah sebagai berikut :

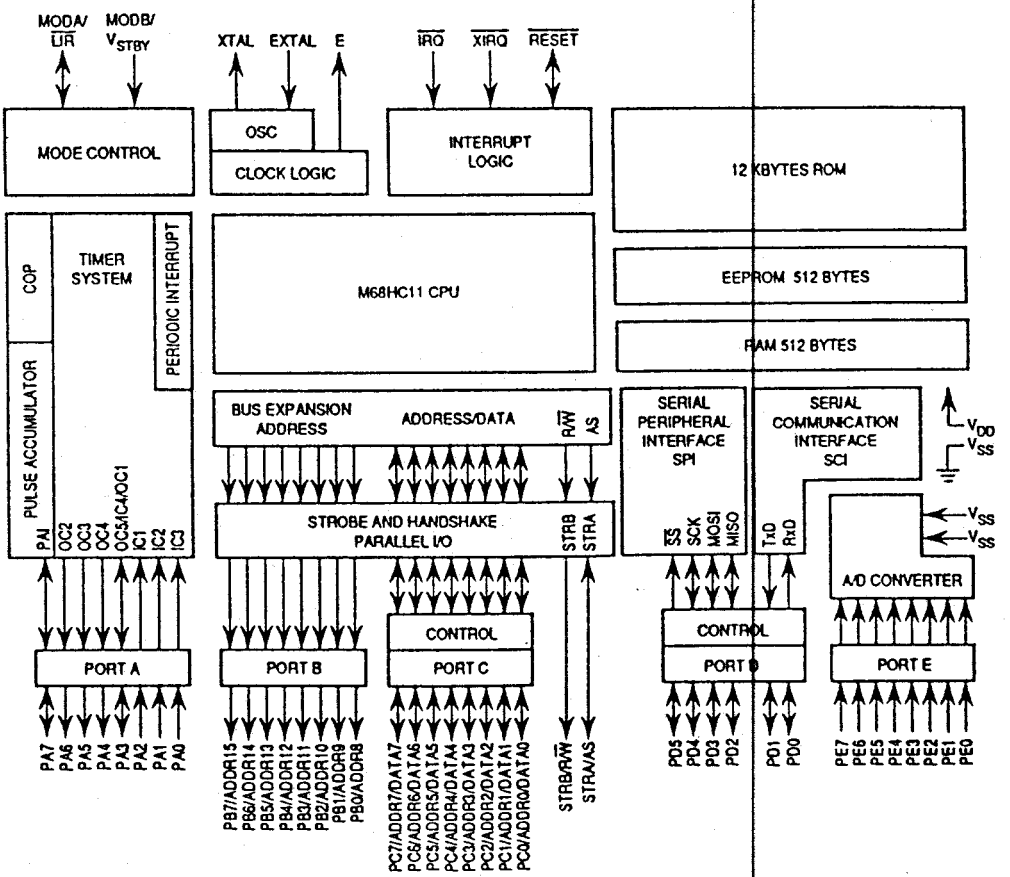
1. Jangka garis adalah tiga kali panjang jangka titik.
2. Luang antara titik-titik dan garis-garis dalam satu huruf adalah sama dengan satu titik.
3. Luang antara huruf-huruf dalam satu kata adalah sepanjang tiga titik.
4. Luang antara kata adalah sepanjang tujuh titik.

II.3 MIKROKONTROLER MC68HC11E9

Mikrokontroler MC68HC11E9 merupakan salah satu anggota keluarga mikrokontroler MC68HC11. Mikrokontroler ini adalah mikrokontroler 8 bit yang dikemas dalam sebuah chip dengan teknologi High Density CMOS (HCMOS) sehingga chip ini mempunyai ukuran yang kecil dengan kemampuan kecepatan akses bus sampai 2 MHz dengan konsumsi suplai yang rendah.

Chip ini mempunyai keistimewaan antara lain :

- Memori EEPROM (Electrically Erasable Programmable Read Only Memory) sebesar 512 byte
- Memori RAM sebesar 512 byte
- Sistem timer 16 bit
- Rangkaian akumulator pulsa 8 bit



Gambar 2.1[1]

Blok diagram MC68HC11E9

---

1 . . . . . , MC68HC11E9 Technical Data, Motorola Inc. Phoenix Arizona, USA, 1991, p. 1-2

- Interface komunikasi serial (SCI = Serial Communication Interface)
- Interface peralatan secara serial (SPI = Serial Peripheral Interface)
- 8 channel Analog to Digital Converter 8 bit
- Rangkaian Real Time Interrupt
- Sistem Watchdog

Selain keistimewaan yang tersebut di atas di dalam chip ini terdapat rangkaian self monitoring untuk mencegah sistem error dan juga sistem clock monitor akan mereset chip secara otomatis apabila clock eksternal hilang atau terlalu lambat. Blok diagram dari chip MC68HC11E9 ini adalah seperti gambar 2.1 berikut.

II.3.1 Central Processing Unit

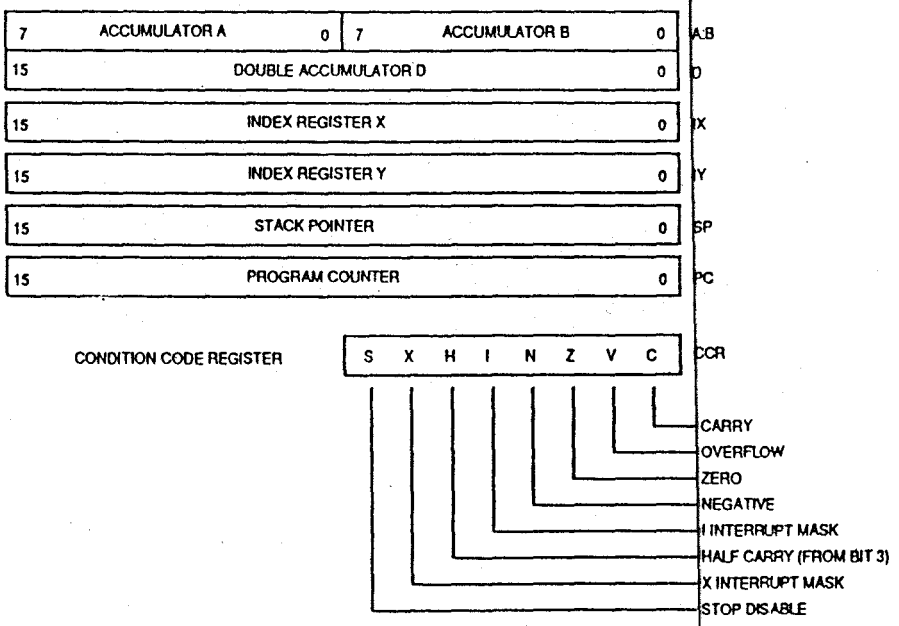
Central Processing Unit (CPU) dari mikrokontroler MC68HC11E9 didesain untuk menangani semua peripheralnya, I/O, dan lokasi memori yang ada dalam jangkauan 64 Kbyte. Tidak ada pemisahan dalam mengakses memori dan port, jadi semua port diakses seperti memori. CPU mikrokontroler MC68HC11E9 memiliki 7 register yang tidak diakses seperti memori. Ketujuh register tersebut adalah : akumulator A dan B, double akumulator D yang merupakan gabungan dari akumulator A dan B, Indeks Register X (IX) dan Y (IY), Stack Pointer (SP), Program Counter (PC), serta Condition Code Register (CCR) yang berisi flag-flag.

CPU MC68HC11E9 mengenal 4 jenis data, yaitu :

1. Data bit
2. 8 bit dan 16 bit signed dan unsigned integer
3. 8 bit unsigned fraction
4. 16 bit address

CPU MC68HC11E9 adalah 8 bit. Satu byte adalah 8 bit yang dapat diakses pada setiap lokasi byte, sedangkan satu word terdiri dari dua byte dengan byte yang berorde tinggi pada address yang berharga rendah.

Keluarga mikrokontroler MC68HC11 menggunakan opcode 8 bit. Setiap opcode mengidentifikasikan maksud dari



Gambar 2.2[2]

Register-register pada chip MC68HC11E9

[2] ibid, p. 3-1

instruksi. Tersedia 256 opcode dan dibagi menjadi 4 halaman pemetaan opcode. Untuk menambah jumlah instruksi maka terdapat byte tambahan yang dinamakan prebyte, seningga jumlah instruksi menjadi bertambah.

Instruksi yang lengkap terdiri dari sebuah opcode, satu, dua, tiga atau tidak ada sama sekali operand dan juga prebyte jika instruksi tersebut menggunakan prebyte. Operand mengandung informasi yang diperlukan CPU untuk melaksanakan instruksi. Instruksi yang lengkap dapat sepanjang satu sampai lima byte.

Pada mikrokontroler ini terdapat 6 mode pengalamatan, yaitu : immediate, direct, extended, index, inheren, dan relatif. Semua mode di atas kecuali mode inheren menggunakan alamat efektif, yaitu alamat memori dimana suatu argumen diambil atau diletakkan. Alamat efektif ini dapat berupa angka atau perhitungan.

II.3.2 Deskripsi Sinyal Pin MC68HC11

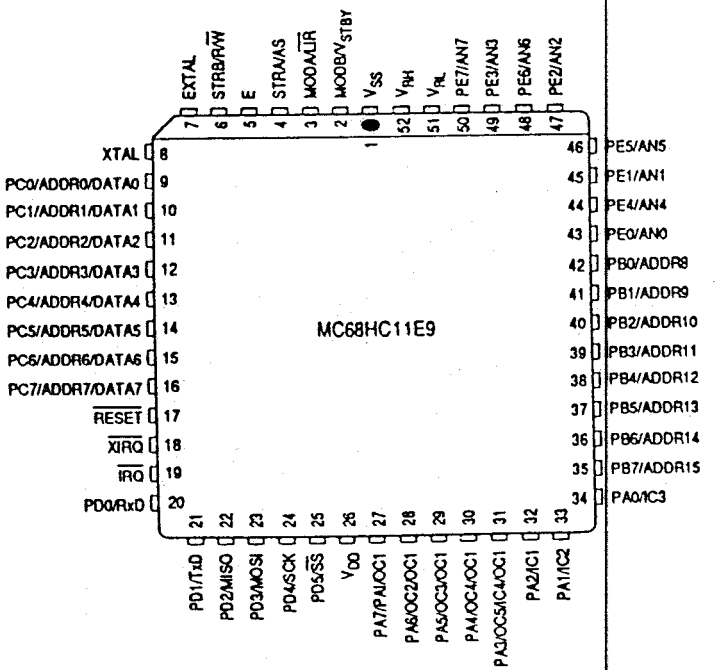
Chip MC68HC11E9 terdiri dari 52 pin yang dikemas secara quad pack seperti yang ditunjukkan dalam gambar 2.3.

A. Power Input (Vdd) dan Ground (Vss)

Pin Vdd dan Vss digunakan untuk suplai daya dari mikrokontroler ini. Vdd adalah input positip suplai daya, sedangkan Vss adalah ground. Mikrokontroler bekerja pada suplai tegangan 5 Volt.

B. Reset ($\overline{\text{RESET}}$)

Pin ini digunakan untuk kontrol sinyal secara bidirectional atau dua arah, yakni sebagai input atau output. Sebagai input (aktif low) maka pin ini digunakan untuk menginisialisasi kondisi awal/startup MC68HC11E9. Sebagai output maka pin ini berfungsi untuk menunjukkan bahwa terdapat suatu 'kegagalan' di dalam mikrokontroler yang dideteksi oleh rangkaian watchdog.



Gambar 2.3[3]

Konfigurasi pin-pin MC68HC11E9

[3] ibid, p. 2-1

C. Crystal Driver (XTAL) dan Input Clock Eksternal (EXTAL)

Dua pin ini dihubungkan ke sebuah kristal atau sebuah rangkaian osilator dari CMOS untuk mengontrol rangkaian pembangkit clock internal. Frekuensi clocknya adalah empat kali frekuensi yang dikeluarkan oleh pin E Clock Output. Jika digunakan sebuah osilator dari CMOS maka osilator tersebut dihubungkan ke pin EXTAL, sedangkan pin XTAL dibiarkan tanpa hubungan atau dihubungkan dengan sebuah resistor antara 10K sampai 100K ke ground untuk mengurangi noise yang timbul.

D. E Clock Output

Pin ini akan mengeluarkan sinyal clock yang dibangkitkan oleh mikrokontroler dengan frekuensi seperempat frekuensi clock yang masuk ke pin EXTAL. Pada saat kondisi E clock output low berarti proses sedang berlangsung di dalam mikrokontroler, bila dalam kondisi high berarti datta masuk ke mikrokontroler.

E. Interrupt Request ($\overline{IRQ}$)

Pin ini digunakan untuk mendeteksi adanya permintaan interrupt asinkron ke MCU (Mikrokontroler Unit). Dengan memanfaatkan 'Option Register' melalui program yang dibuat maka pin ini dapat dipilih untuk sinyal interrupt negative edge sensitive triggering atau level sensitive triggering. Bila digunakan untuk mereset MCU maka pin ini dikonfigurasi level sensitive triggering. Pin ini dihubungkan dengan

resistor pullup 4,7K ke Vdd.

F. Non Maskable Interrupt ($\overline{XIRQ}$)

Pin ini digunakan untuk mendeteksi adanya permintaan non maskable interrupt, yang artinya MCU harus lebih memprioritaskan pelayanan interrupt ini dibandingkan dengan yang lain. Pada saat MCU direset, maka X bit pada Condition Code Register (CCR) diset dan tidak ada interrupt lain yang dilayani sampai software yang dibuat menonaktifkan X bit tersebut. $\overline{XIRQ}$ ini sering digunakan untuk mendeteksi adanya kehilangan suplai daya.

G. Mode A/Load Instruction Register dan Mode B/Stanby Voltage (MODA/LIR dan MODB/V_{stby})

Pada saat pertama kali MCU direset, maka pin ini digunakan untuk mendeteksi mode operasi yang digunakan MCU.

Tabel 2.1 menunjukkan kondisi bit dari pin-pin ini, yang menunjukkan operasi yang sedang dipergunakan MCU.

Setelah mode operasi MCU terpilih, pin LIR menjadi output open drain untuk menunjukkan bahwa instruksi mulai dikerjakan. LIR menjadi low selama siklus E clock yang pertama untuk setiap instruksi yang dikerjakan. Sinyal V_{stby} digunakan sebagai input tegangan suplai RAM sesaat. Jika tegangan pada pin ini lebih dari tegangan ambang MOS (0,7 volt) di atas tegangan Vdd, RAM 512 byte di dalam MCU akan disuplai oleh tegangan dari pin ini. Hal ini akan menahan isi dari RAM tanpa perlu Vdd menyuplai MCU.

Tabel 2.1

Mode Operasi MCU

MODA	MODB	Mode Operasi
1	0	Single Chip
1	1	Expanded Multiplexed
0	0	Special Bootstrap
0	1	Special Test

H. Tegangan Referensi Konverter A/D (VRL dan VRH)

Dua pin ini digunakan sebagai input tegangan referensi ADC di dalam MCU. VRL untuk tegangan referensi rendah, yaitu 0 Vdc. VRH untuk tegangan referensi tinggi. Tegangan minimum sebesar 2,5 V dan maksimum sebesar tegangan Vdd.

I. Strobe B dan Read/Write (STRB/R/$\overline{W}$)

Dua pin ini digunakan untuk menunjukkan atau mengatur arah bus data tergantung dari mode operasi yang dipergunakan.

Pada mode operasi single chip output STRB dipakai sebagai programmable strobe untuk sinyal handshaking pada peralatan I/O lain yang dihubungkan ke MCU secara paralel.

Pada mode operasi expanded multiplexed pin R/$\overline{W}$ digunakan untuk mengontrol arah transfer data dari bus data. Jika kondisi R/$\overline{W}$ rendah berarti data sedang dikirim/ditulis ke bus data. Jika kondisi R/$\overline{W}$ tinggi berarti data pada bus

data sedang dibaca oleh MCU. R/$\overline{W}$ akan tetap dalam kondisi rendah selama penulisan sekelompok data seperti data double byte ke bus data.

J. Strobe A dan Address Strobe (STRA/AS)

Pin ini memiliki dua fungsi yang terpisah sifatnya tergantung dari mode operasi yang sedang digunakan MCU. Pada mode single chip, STRA berfungsi sebagai strobe input yang dapat diprogram untuk handshaking peralatan I/O luar yang disambungkan ke MCU secara paralel. Pada mode operasi expanded multiplexed pin ini (AS) berfungsi sebagai output, digunakan untuk demultipleksing alamat dan data pada port C.

K. Sinyal Port

MCU MC68HC11E9 mempunyai lima buah prot yang masing-masing mempunyai 8 buah bit (tiap pin 1 bit). Port A, D, dan E tidak tergantung dari mode operasi tetapi port B dan C dipengaruhi oleh mode operasi. Port B akan menghasilkan sinyal output yang berfungsi secara umum pada mode operasi single chip.

Jika mikrokontroler pada mode expanded multiplexed, port B merupakan address line yang berkondisi tinggi. Port C merupakan port input/output secara umum jika MCU pada mode operasi single chip. Jika MCU pada mode operasi expanded multiplexed, port C digunakan untuk multiplexed address atau data bus.

Tabel 2.2 menunjukkan secara terperinci fungsi tiap pin

dari port mikrokontroler untuk setiap mode operasi.

a. Port A

Pada semua mode operasi, port A dapat digunakan sebagai empat timer Input Capture (IC) dan empat Output Compare (OC) atau untuk empat Output Compare, tiga Input Capture dan satu input pulsa akumulator atau untuk tiga Input Capture dan lima Output Compare, Setiap pin pada port A yang tidak difungsikan sebagai timer dapat dipakai sebagai general purpose input/output.

b. Port B

Selama operasi dalam mode single chip, semua pin pada port B berfungsi untuk general purpose output. Port B juga dapat dipakai untuk strobed output dimana pulsa strobe output akan muncul pada pin STRB setiap kali data akan dikeluarkan melalui port B.

Pada mode operasi expanded multiplexed, semua pin pada port B beraksi sebagai sinyal output untuk address. Selama siklus MCU, bit 15 sampai 8 dari address bus dikeluarkan melalui PB7 - PB0.

c. Port C

Pada mode operasi single chip, semua pin pada port C berfungsi untuk general purpose input/output. Data input pada port C dapat tersimpan sementara pada register dengan

Tabel 2.2[4]

Fungsi Port Tiap Bit

Port/Bit	Single-Chip and Bootstrap Mode	Expanded Multiplexed and Special Test Mode
PA0	PA0/IC3	
PA1	PA1/IC2	
PA2	PA2/IC1	
PA3	PA3/OC5/IC4/and-or OC1	
PA4	PA4/OC4/and-or OC1	
PA5	PA5/OC3/and-or OC1	
PA6	PA6/OC2/and-or OC1	
PA7	PA7/PAI/and-or OC1	
PB0	PB0	ADDR8
PB1	PB1	ADDR9
PB2	PB2	ADDR10
PB3	PB3	ADDR11
PB4	PB4	ADDR12
PB5	PB5	ADDR13
PB6	PB6	ADDR14
PB7	PB7	ADDR15
PC0	PC0	ADDR0/DATA0
PC1	PC1	ADDR1/DATA1
PC2	PC2	ADDR2/DATA2
PC3	PC3	ADDR3/DATA3
PC4	PC4	ADDR4/DATA4
PC5	PC5	ADDR5/DATA5
PC6	PC6	ADDR6/DATA6
PC7	PC7	ADDR7/DATA7
PD0	PD0/RxD	
PD1	PD1/TxD	
PD2	PD2/MISO	
PD3	PD3/MOSI	
PD4	PD4/SCK	
PD5	PD5/$\overline{SS}$	
—	STRA	AS
—	STRB	R/$\overline{W}$
PE0	PE0/AN0	
PE1	PE1/AN1	
PE2	PE3/AN2	
PE3	PE3/AN3	
PE4	PE4/AN4	
PE5	PE5/AN5	
PE6	PE6/AN6	
PE7	PE7/AN7	

[4] ibid, p. 2-8

memanfaatkan pin STRA. Port C juga digunakan untuk handshaking lintas data I/O dimana pin input STRBA dan pin output STRB digunakan untuk control handshaking.

Pada mode operasi multiplexed, semua pin pada port C dapat dikonfigurasikan untuk bermacam-macam sinyal data atau address. Pada siklus clock MCU untuk pengalamatan / addressing, bit 0 - 7 yang digunakan untuuk address dikeluarkan melalui PC0 - PC7. Pada siklus clock MCU untuk data (E HIGH), pin-pin PC0 - PC7 merupakan jalur data dua arah atau bidirectional. Arah data pada port C ditujukan dengan sinyal dari pin R/$\overline{W}$.

d. Port D

Pin PD0 - PD7 dapat digunakan sebagai sinyal input / output. Pin-pin ini juga digunakan untuk sinyal-sinyal komunikasi serial interface (SCI) dan komunikasi peralatan secara serial interface (SPI).

Pin PD0 adalah sinyal data yang diterima (RxD) untuk SCI, pin PD1 adalah sinyal data yang dikeluarkan (TxD) untuk SCI, pin PD2 - PD5 digunakan untuk SPI. PD2 adalah sinyal Master In Slave Out (MISO). PD3 adalah sinyal Master Out Sinyal In (MOSI). PD4 adalah sinyal clock serial (SCK) dan PD5 digunakan untuk memilih sinyal Slave ($S\overline{S}$).

e. Port E

Port E digunakan untuk sinyal input data atau digunakan untuk sinyal analog to digital convertion sebanyak 8

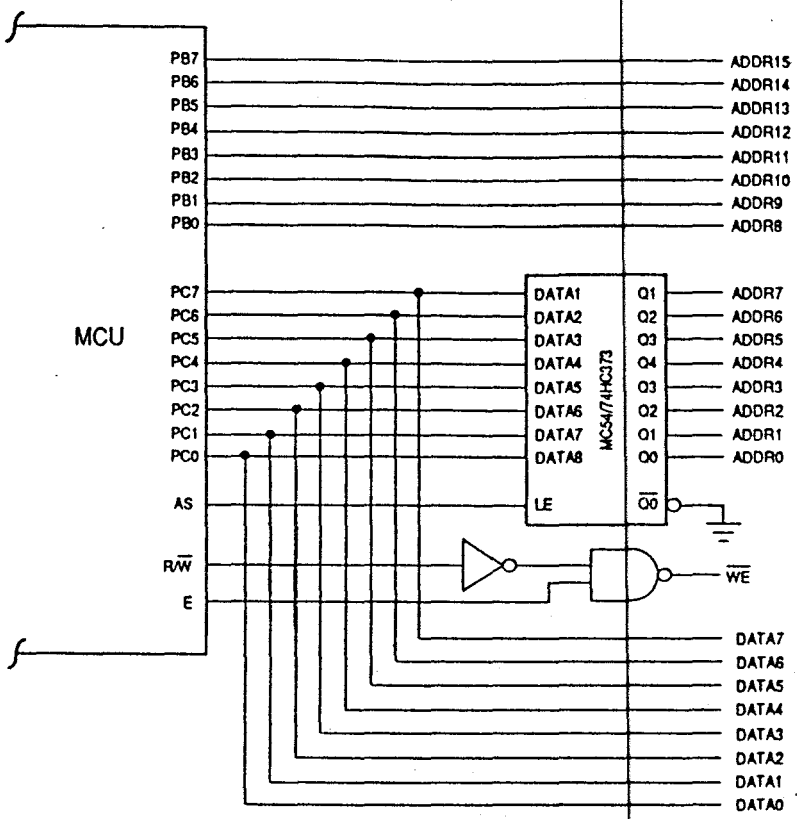
channel. Pembacaan data pada port E selama proses sampling data untuk konversi analog to digital sedang berlangsung dapat menyebabkan data hasil konversi tidak valid. Oleh karena itu dihindarkan proses pembacaan data pada port E selama siklus konversi sedang berlangsung.

II.3.3 Mode Operasi MC68HC11E9

Terdapat empat mode operasi yang digunakan pada MC68HC11E9, yakni : single chip, expanded multiplexed, special bootstrap, dan special test. Single chip dan expanded multiplexed adalah mode yang biasa digunakan. Pada mode single chip, hanya menggunakan memori di dalam chip MC68HC11E9. Sedangkan pada mode expanded multiplexed dapat menggunakan memori dari luar selain dari dalam chip MC68HC11E9 sendiri. Mode special bootstrap merupakan variasi dari mode single chip, yaitu mode khusus yang menggunakan bootloader dari program ROM bootstrap. Mode special test merupakan mode khusus yang digunakan oleh pabrik pembuat chip untuk mentes chip MC68HC11E9.

A. Single Chip

Pada mode operasi ini, MC68HC11E9 berfungsi sebagai mikrokontroler tanpa address atau data bus dari luar. Port B, C, strobe A dan B berfungsi sebagai general purpose input / output dan sinyal handshake.



Gambar 2.4[5]

Sistem Bus Ekspansi

B. Expanded Multiplexed

Pada mode operasi ini MC68HC11E9 mempunyai kemampuan untuk mengakses 64K byte ruang address. Jumlah ini adalah sama dengan address memori pada chip yang digunakan untuk mode operasi single chip ditambah dengan address untuk peralatan dan memori dari luar. Bus ekspansi dibuat pada port B dan C serta sinyal kontrol AS dan R/W. Gambar 2.4

[5] ibid, p. 4-2

menunjukkan cara untuk menambah bus ekspansi untuk pengalamatan device atau memori dari luar melalui port C dan B.

C. Special Bootstrap

Bila MCU pada mode ini, maka setelah reset MCU akan mengaktifkan address $BF00 - $BFFF yang berisi program bootloader yang termuat di dalam ROM bootstrap.

D. Special Test

Mode ini khusus digunakan untuk menguji chip MC68HC11E9. Jadi hanya digunakan oleh pabrik pembuat chip ini. Pada mode ini reset dan interrupt vektor diset pada address $BFFF - $BFC0.

II.3.4 Peta Memori MC68HC11E9

Komposisi peta memori untuk tiap mode operasi adalah seperti ditunjukkan pada gambar 2.5. Dari gambar tersebut terlihat bahwa pada memori untuk mode single chip dan expanded multiplexed hampir sama, hanya untuk single chip eksternal memori tidak digunakan. Sedang untuk expanded multiplexed digunakan memori eksternal yang besarnya dapat sampai kurang lebih 64 Kbyte. Apabila terdapat pemakaian alamat yang sama antara RAM eksternal dan memori internal serta register internal, maka digunakan urutan prioritas hardware sebagai berikut : Register, RAM, dan ROM.

RAM internal 512 byte adalah RAM statis yang bersifat

menyimpan data, variabel dan instruksi sementara. ROM bootstrap 256 byte adalah ROM internal yang memuat program bootloader yang akan muncul pada address $BF00 - $BFFF bila MCU berada pada mode special bootstrap. 12 Kbyte ROM berada pada lokasi $D000 - $FFFF untuk semua mode. 512 byte EEPROM berada pada lokasi $B600 - $B7FF untuk semua mode. EEPROM dapat diprogram dan dihapus isi memorinya oleh software. ROM dan EEPROM dapat aktif atau tidak aktif dengan cara mengatur control bit pada CONFIG register. CONFIG register adalah salah satu dari register internal 64 byte yang terletak pada lokasi memori $1000 - $103F yang digunakan untuk mengontrol operasi MCU.

Keempat mode operasi yang telah dijelaskan di atas dipilih melalui pin MODA dan MODB. Kondisi bit kedua pin ini menentukan jenis mode operasi yang dipilih dan juga menentukan kondisi bit pada priority dan Mode Select Register (HPRIO). Kondisi bit yang dipengaruhi oleh mode operasi yang dipilih ini adalah untuk bit RBOOT, SMOD, MDA, dan IRVNE. Secara ringkas kondisi bit-bit ini untuk tiap mode operasi adalah seperti ditunjukkan pada tabel 2.3.

Kondisi bit control RBOOT akan mempengaruhi apakah bootloader ROM aktif pada address $BF00 - $BFFF atau tidak. Kondisi bit SMOD akan mempengaruhi apakah MCU pada mode normal atau mode special.

Kondisi bit MDA akan mempengaruhi MCU apakah MCU menggunakan memori eksternal atau tidak. Kondisi bit IRVNE

Tabel 2.3[6]

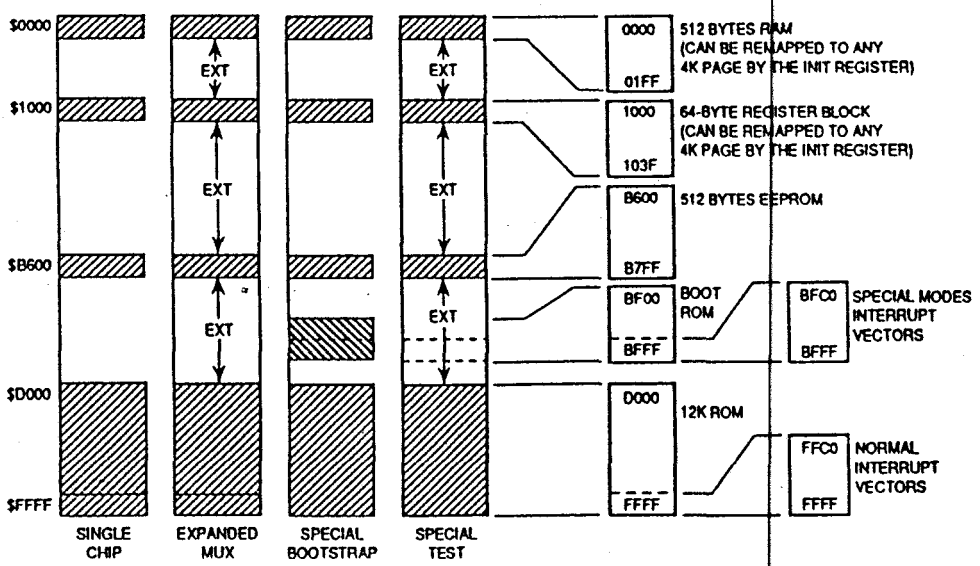
Kondisi Bit Register HPRIO

pada tiap mode operasi

Inputs		Mode	Control Bits in HPRIO (Latched at Reset)		
MODB	MODA		RBOOT	SMOD	MDA
1	0	Single Chip	0	0	0
1	1	Expanded Multiplexed	0	0	1
0	0	Special Bootstrap	1	1	0
0	1	Special Test	0	1	1

HPRIO — Highest Priority I-Bit Interrupt and Miscellaneous **$103C**

	Bit 7	6	5	4	3	2	1	Bit 0
	RBOOT	SMOD	MDA	IRVNE	PSEL3	PSEL2	PSEL1	PSEL0
RESET:	—	—	—	—	0	1	0	1



Gambar 2.5[7]

Peta Memori MC68HC11E9

[6] ibid, p. 4-8

[7] ibid, p. 4-4

akan mempengaruhi MCU apakah MCU dapat melakukan pembacaan data pada bus eksternal atau tidak.

II.3.5 Analog to Digital Converter

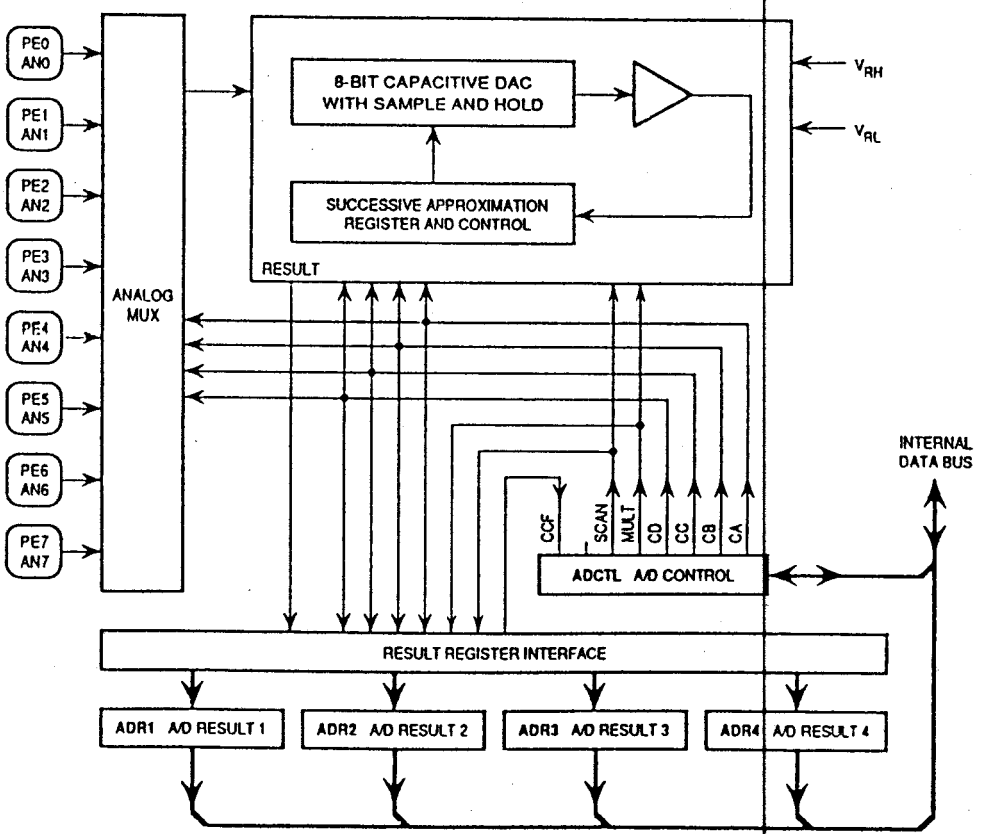
Untuk mengaktifkan A/D Converter di dalam MCU adalah dengan menset bit ADPU (bit ke-7) pada register OPTION dalam logika 1. Analog to digital converter di dalam MCU adalah 8 bit A/D Converter dari jenis successive aproximation sebanyak 8 channel. A/D Converter ini menggunakan rangkaian sample dan hold untuk mengurangi error konversi karena cepatnya perubahan sinyal input. Pin VRL dan VRH digunakan untuk tegangan referensi dengan tegangan pada VRL sama dengan tegangan Vss (0 volt) sedangkan tegangan VRH dapat sampai sebesar tegangan Vdd (2,5 - 5 volt). 8 bit A/D Converter pada MCU ini mempunyai tingkat akurasi sampai 1 bit LSB.

Secara sistem, A/D Converter pada MCU ini terdiri dari empat blok fungsional, yaitu multiplexer, analog converter, digital control, dan result storage. Blok diagram sistem A/D Converter ini adalah seperti ditunjukkan dalam gambar 2.6..

Multiplexer akan memilih salah satu dari 16 sinyal input analog dengan cara melalui bit control CD - CA pada register ADCTL. 8 sinyal input analog dari 16 channel input ini berasal dari port E, 4 sinyal input analog lainnya dihubungkan ke sinyal tegangan referensi untuk maksud pengetesan, dan sisanya adalah cadangan.

Proses konversi dari analog ke digital dilakukan pada

analog converter. Bagian ini terdiri dari Digital to Analog Converter (DAC), Sample and Hold, komparator, dan Successive Approximation Register (SAR). Pertama kali SAR akan menset harga MSB pada logika 1 dan yang lainnya 0. Kemudian oleh DAC dikonversikan ke analog dan hasil konversi ini akan dibandingkan dengan harga sinyal analog yang ditangkap oleh rangkaian Sample and Hold. Bila harga sinyal analog hasil



Gambar 2.6[8]

Blok Diagram Analog to Digital Converter

[8] ibid, p. 10-2

konversi DAC lebih besar maka komparator akan menghasilkan level tegangan logika 0 dan jika lebih kecil komparator akan menghasilkan level tegangan logika 1. Hasil dari komparator akan dikirimkan ke MSB dari SAR (bit ke-8) dan SAR akan menggeser isi bitnya ke arah LSB. Kemudian DAC akan mengkonversikan lagi harga digital pada SAR ke analog dan membandingkannya lagi. Demikian proses ini berlangsung terus sampai bit yang terakhir dari SAR telah selesai dibandingkan.

Hasil proses konversi pada SAR kemudian dimasukkan ke dalam result register yang terdiri dari 4 buah register ADR1 - ADR4 yang terletak pada lokasi memori $1031 - $1034. Semua operasi A/D Converter dikontrol oleh Digital Control yang merupakan register control ADCTL yang terletak pada lokasi memori $1030. Register ADCTL adalah seperti ditunjukkan pada gambar 2.7.

Dalam satu kali operasi A/D Converter melakukan 4 kali konversi sinyal analog yang masuk dan memasukkan hasil konversi ke result register secara berurutan dari ADR1 ke ADR4. Setelah hasil konversi keempat pada ADR4 maka bit CCF pada register ADCTL diset 1.

Terdapat dua jenis operasi untuk A/D Converter ini, yaitu single channel dan multi channel. Bila bit MULT pada register ADCTL diset 0 berarti operasi single channel yang dipilih dan bila diset 1 berarti operasi multiple channel yang dipilih.

Bit CD0 - CD3 digunakan untuk memilih channel yang

digunakan untuk input sinyal analog. Tabel 2.4 menunjukkan daftar channel untuk tiap kondisi dari bit CD0 - CD3.

ADCTL — A/D Control/Status **$1030**

	Bit 7	6	5	4	3	2	1	Bit 0
	CCF	0	SCAN	MULT	CD	CC	CB	CA
RESET:	0	0	U	U	U	U	U	U

Gambar 2.7[9]

Register ADCTL

Tabel 2.4

Daftar Channel A/D Converter

Channel Select Control Bits CD:CC:CB:CA	Channel Signal	Result in ADRx if MULT=1
0000	AN0	ADR1
0001	AN1	ADR2
0010	AN2	ADR3
0011	AN3	ADR4
0100	AN4	ADR1
0101	AN5	ADR2
0110	AN6	ADR3
0111	AN7	ADR4
10xx	Reserved	--
1100	VRH^*	ADR1
1101	VRL^*	ADR2
1110	$(VRH)/2^*$	ADR3
1111	Reserved*	ADR4

Dalam mode single channel, bila bit SCAN = 0, maka A/D Converter akan mengkonversikan sinyal analog yang masuk

[9] ibid, p. 10-8

melalui channel yang dipilih selama empat kali dan menyimpan hasilnya ke empat buah result register (ADR1 - ADR4). Bila proses konversi keempat telah selesai, maka proses konversi akan berhenti menunggu perintah baru yang ditulis ke register ADCTL. Sedangkan bila bit SCAN = 1, proses konversi tidak berhenti sampai konversi keempat tapi berlanjut terus dengan penyimpanan hasil konversi diulang mulai register ADR1, kemudian ADR2, dan seterusnya.

Dalam mode multiple channel, bit CB dan CA pada register ADCTL tidak mempunyai arti, seleksi channel dilakukan dengan bit CC dan CD. Jadi tiap kondisi gabungan CC dan CD terdapat empat buah channel. Jika bit SCAN = 0, maka dalam satu kali operasi, A/D Converter akan mengkonversi empat buah sinyal analog dari empat channel yang dipilih. Proses akan berhenti setelah konversi keempat dan menunggu perintah baru ditulis pada register ADCTL. Bila bit SCAN = 1, maka proses konversi akan terus berlanjut dengan menyimpan hasil konversi kelima ke register ADR1, hasil konversi keenam ke register ADR2, dan seterusnya.

II.3.6 Serial Communication Interface (SCI)

Fasilitas Serial Communication Interface yang disediakan oleh MCU MC68HC11E9 adalah dari jenis universal asynchronous receiver transmitter (UART). Tersedia format NRZ (satu bit start, delapan atau sembilan data bit, dan satu stop bit) dan berbagai variasi dari baud rate. Fungsi transmiter dan receiver dari SCI tidak saling tergantung,

tetapi menggunakan data format dan bit rate yang sama.

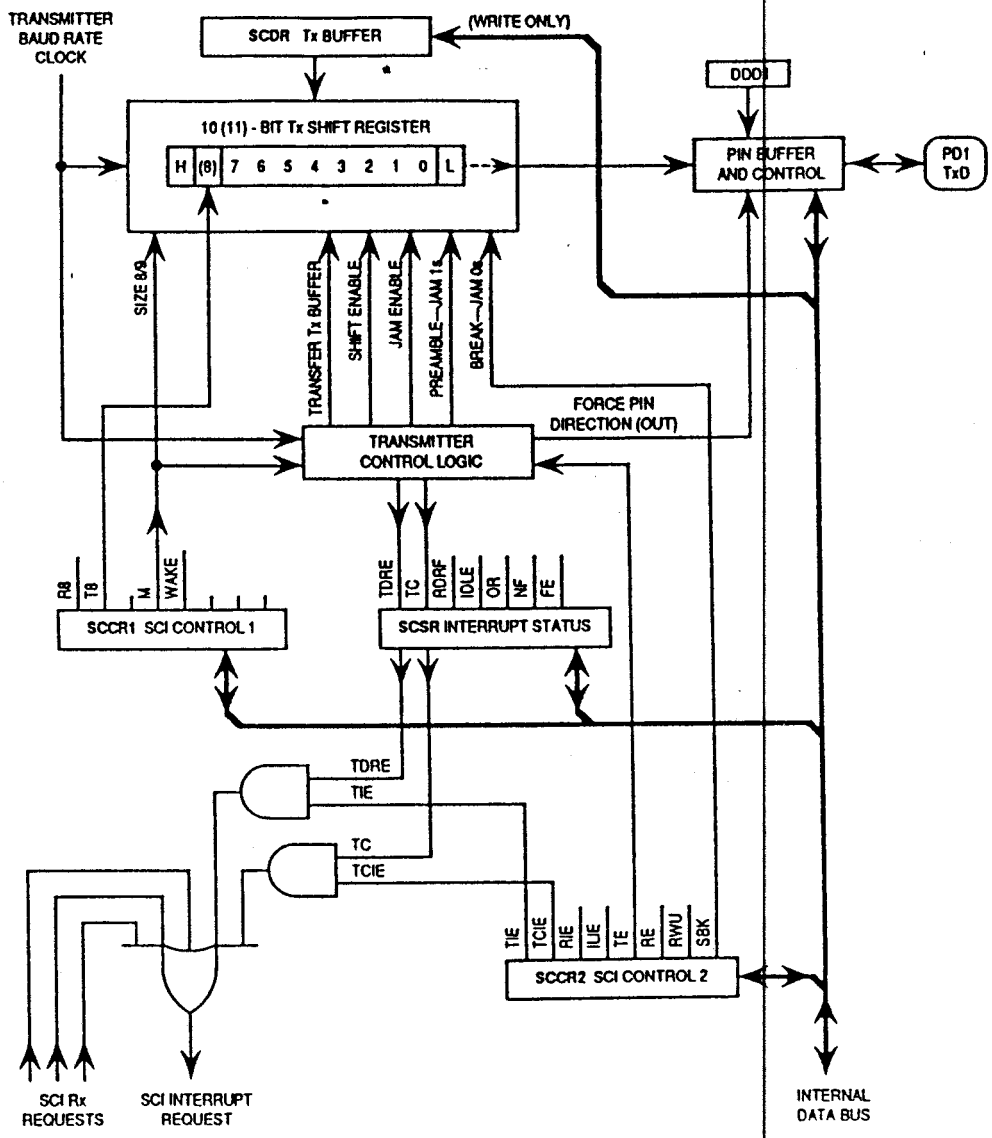
Format data untuk komunikasi serial ini harus memenuhi kriteria seperti berikut ini :

1. Kondisi idle line adalah kondisi logika 1 sebelum pengiriman atau penerimaan data / karakter.
2. Start bit, kondisi logika 0, menunjukkan awal data atau karakter yang dikirim / diterima.
3. Bit data pertama yang dikirim / diterima adalah least significant bit (LSB).
4. Stop bit, kondisi logika 1, menunjukkan akhir dari satu frame data. Satu frame data terdiri dari satu start bit, delapan atau sembilan bit data, dan satu stop bit.
5. Kondisi break didefinisikan sebagai pengiriman atau penerimaan data berkondisi logika 0 sedikitnya sepanjang satu frame.

Transmiter SCI dapat menghasilkan sinyal idle line dan break. Blok diagram untuk transmiter SCI adalah seperti pada gambar 2.8.

Jantung dari Transmiter SCI adalah pada serial transmit shift register. Shift register ini mengambil data dari transmit buffer. Data akan masuk ke dalam buffer apabila software yang dibuat melakukan penulisan data ke SCI data register (SCDR). Ketika data ditransfer dari buffer ke shift register, start bit berkondisi logika 0 yang pertama ditransfer, kemudian baru delapan atau sembilan bit data, dan terakhir adalah stop bit.

Satu frame data akan dikeluarkan ke pin TxD jika bit TE



Gambar 2.8[10]

Blok Diagram Transmiter SCI MC68HC11E9

10 ibid, p. 7-2

pada serial communication control register 2 (SCCR2) pada kondisi logika 1. Bit T8 pada SCI control register 1 (SCCR1) berlaku sebagai bit ke-9 pada format data / karakter.

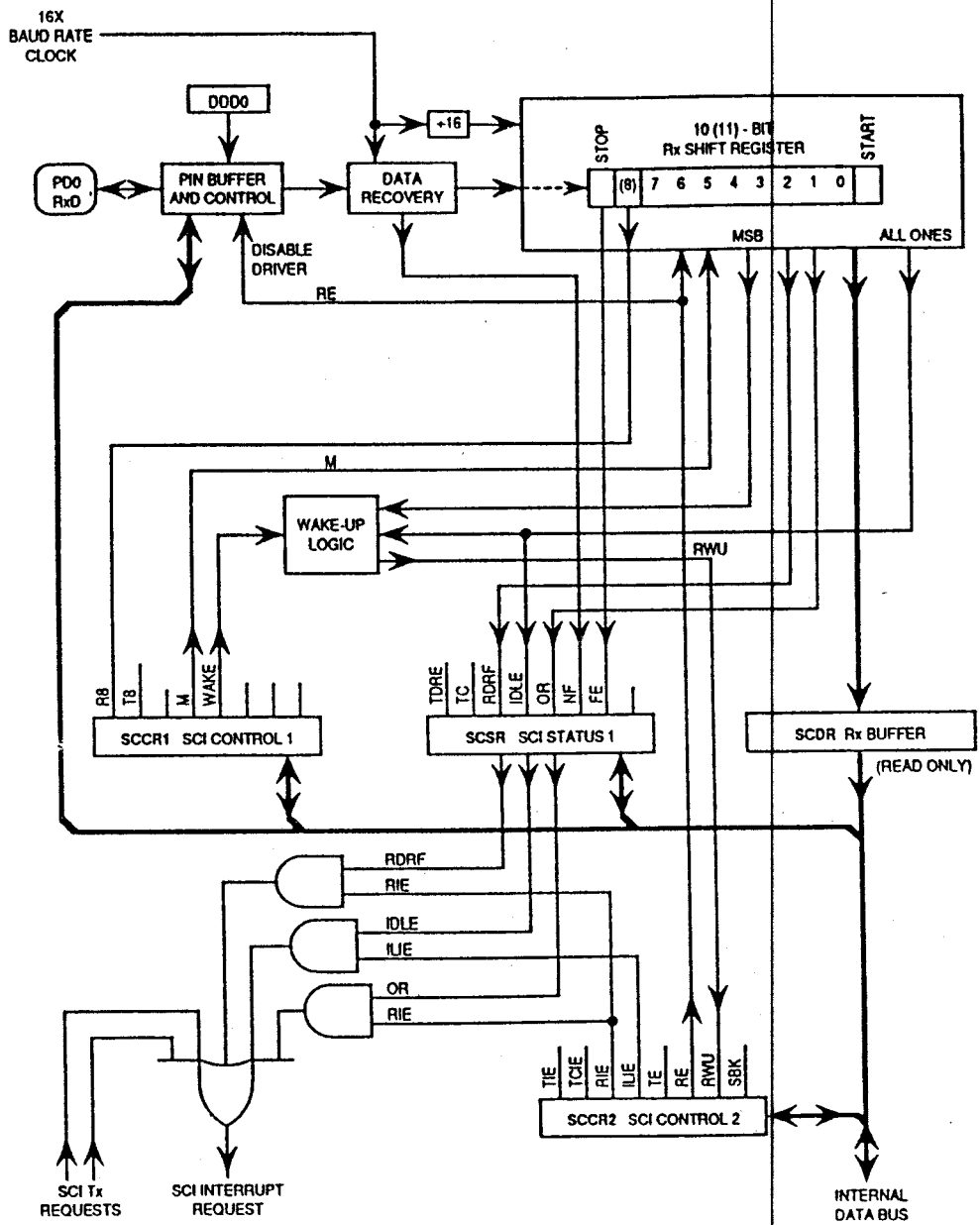
Bit ini digunakan hanya jika bit M pada SCCR1 pada kondisi logika 1 yang berarti data yang dikirim / diterima adalah 9 bit. Bit TDRE dan TC secara otomatis diset oleh transmitter control logic. Kedua bit ini dapat dibaca melalui software. Bit-bit transmit interrupt enable (TIE), transmit complete interrupt enable (TCIE). TDRE dan TC digunakan untuk membangkitkan sinyal SCI interrupt request.

Pada operasi penerimaan data, maka operasinya adalah kebalikan dari operasi pengiriman data. Blok diagram receiver SCI adalah seperti ditunjukkan pada gambar 2.9.

Jantung dari receiver SCI adalah serial receive shift register. Shift register ini diaktifkan oleh bit receive enable (RE) pada SCI control register 2 (SCCR2). Jika bit ini pada kondisi logika 1 maka data yang diterima melalui pin Rx yang terletak pada buffer akan ditransfer ke shift register.

Bit M pada register SCCR1 menentukan apakah shift register akan menerima sepanjang 10 atau 11 bit. Setelah stop bit terdeteksi maka data yang diterima ditransfer ke SCDR dan receive data register full status flag (RDRF) diset pada kondisi logika 1.

Ketika data / karakter siap ditransfer ke buffer sedangkan data / karakter yang terdahulu belum terbaca, maka terjadi kondisi overrun.



Gambar 2.9[11]

Blok Diagram Receiver SCI MC68HC11E9

[11] ibid, p. 7-4

Pada kondisi ini data tidak akan ditransfer dan bit overrun status flag (OR) diset pada kondisi logika 1 untuk menunjukkan adanya kesalahan.

Bit control WAKE dari register SCCR1 digunakan untuk memilih apakah menggunakan sinyal MSB (Address mark) atau sinyal idle line untuk mengaktifkan receiver. Jika kondisi yang dipilih untuk mengaktifkan receiver terdeteksi maka wake up logic akan menset bit RWU pada register SCCR2 pada kondisi logika 0 yang akan mengaktifkan receiver.

Sistem SCI dikontrol oleh lima buah register, yaitu register BAUD ($102B), SCDR ($102F), SCCR1 ($102C), SCCR2 ($102D), SCSR ($102E).

Register BAUD digunakan untuk memilih baud rate untuk operasi SCI. Register SCCR1 mengandung bit kontrol untuk menentukan panjang format data dan menyeleksi metode yang digunakan untuk mengaktifkan (wake up) receiver SCI. Register SCCR2 mengandung bit-bit kontrol utama untuk operasi SCI. Empat bit orde tinggi dari register ini digunakan untuk mengontrol sinyal interrupt request. Bit TE dan RE digunakan untuk mengaktifkan transmiter dan receiver.

Bit RWU digunakan untuk mengaktifkan receiver. Bit SBK digunakan untuk membangkitkan sinyal break pada transmiter (TxD). Register SCSR mengandung dua bit status flag untuk transmiter, yaitu bit TDRE dan TC serta lima buah bit status flag untuk receiver, yaitu bit RDRF, OR, idle line detect (IDLE), Noise flag (NF) dan Framing Error indication (FE). SCDR merupakan dua buah register yang terpisah, yaitu

Transmit Data Register (TDR) dan Receive Data Register (RDR). Jika software membaca SCDR berarti RDR yang aktif, jika software menuliskan data pada SCDR berarti TDR yang aktif.

II.3.7 Serial Peripheral Interface (SPI)

Serial Peripheral Interface (SPI) adalah interface sinkronous yang dapat difungsikan untuk menghubungkan atau interkoneksi dengan SPI mikrokontroler lain atau peripheral tipe SPI. Di dalam SPI, data dan clock berada pada jalur-jalur terpisah. Sistem SPI 68HC11 dapat dikonfigurasikan sebagai Master atau sebagai Slave.

A. Deskripsi Sinyal SPI

Terdapat empat sinyal dasar SPI (MISO, MOSI, SCK, dan SS). Setiap sinyal akan dijelaskan pada bagian mode master dan slave.

Setiap jalur output SPI mempunyai bit Data Direction Register yang aktif sesuai dengan jalur yang bersangkutan. Jika bit-bit ini nol atau tidak aktif, jalur yang bersangkutan akan terputus dari fungsi logika SPI dan menjadi jalur input multifungsi.

B. Master In Slave Out (MISO)

Jalur MISO berfungsi sebagai input pada peralatan Master dan sebagai output pada peralatan Slave. Jalur ini adalah jalur transfer data serial pada satu arah, dengan bit

MSB yang dikirim pertama kali. Jalur MISO dari peralatan Slave berada pada kondisi impedansi tinggi jika slave tidak dipilih.

C. Master Out Slave In (MOSI)

Jalur MOSI berfungsi sebagai output pada peralatan Master dan sebagai input pada peralatan slave. Jalur ini adalah jalur transfer data serial pada satu arah, dengan bit MSB yang dikirim pertama kali.

D. Serial Clock (SCK)

Serial clock digunakan untuk sinkronisasi pergerakan data, baik masuk atau keluar dari peralatan melalui jalur MISO atau MOSI. Pada peralatan-peralatan Master dan Slave dimungkinkan untuk terjadi pertukaran satu byte informasi selama proses dalam delapan clock. Karena SCK digerakkan oleh peralatan Master, jalur ini menjadi input pada peralatan Slave.

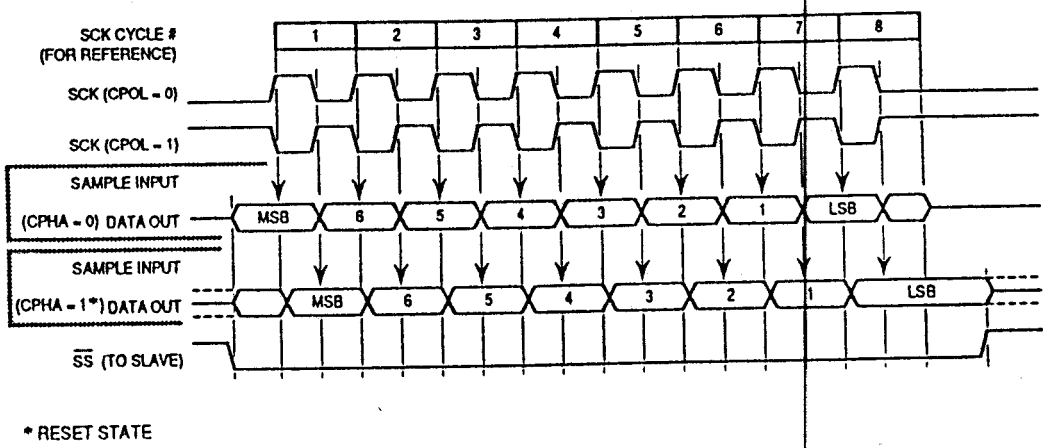
Pada gambar 2.10 dapat dilihat empat kemungkinan model clock atau timing SCK, yang didapat dengan mengeset bit-bit kontrol CPOL dan CPHA pada Serial Peripheral Control Register (SPCR). Baik Master maupun Slave harus beroperasi pada timing yang sama. Peralatan Master selalu menempatkan data pada jalur MOSI setengah gelombang clock sebelum transisi clock SCK, dengan tujuan agar peralatan slave dapat *melatch* data tersebut.

Dua bit kontrol SPRO dan SPRI dalam register SPCR dari

peralatan master memilih clock rate. Pada peralatan Slave, SPRO dan SPRI tidak mempunyai efek pada pengoperasian SPI.

E. Slave Select ($\overline{SS}$)

Jalur input Slave Select ($\overline{SS}$) digunakan untuk memilih peralatan Slave. Pin ini harus pada level rendah saat transaksi data dan harus tetap rendah selama jangka waktu transaksi.



Gambar 2.10[12]

Model SCK dengan setting CPOL dan CPHA

Jalur $\overline{SS}$ ini pada Master harus berlevel tinggi. Jika level turun ke rendah, sebuah Mode Fault Error Flag (MODF) menjadi set dalam Serial Peripheral Status Register (SPSR). Pin $\overline{SS}$ dijadikan output multifungsi dengan memberikan logik

---

[12] ibid, p. 8-3

1 pada bit ke-5 dari port D Data Direction Register (DDDR, sehingga menonaktifkan rangkaian mode fault. Sedangkan ketiga jalur SPI lain berfungsi sebagai SPI saat SPI aktif.

Jika CPHA = 0, pergeseran clock adalah OR dari $\overline{SS}$ dengan SCK. Pada mode phase clock ini, $\overline{SS}$ harus menjadi tinggi di antara deretan karakter-karakter yang dikirim SPI. Jika CPHA = 1, $\overline{SS}$ bisa tetap rendah untuk beberapa karakter SPI. Pada kasus dimana hanya terdapat satu SPI MCU Slave, jalur $\overline{SS}$-nya dapat tetap berlevel Vss selama mode clock CPHA = 1 digunakan.

F. Deskripsi Fungsi Internal

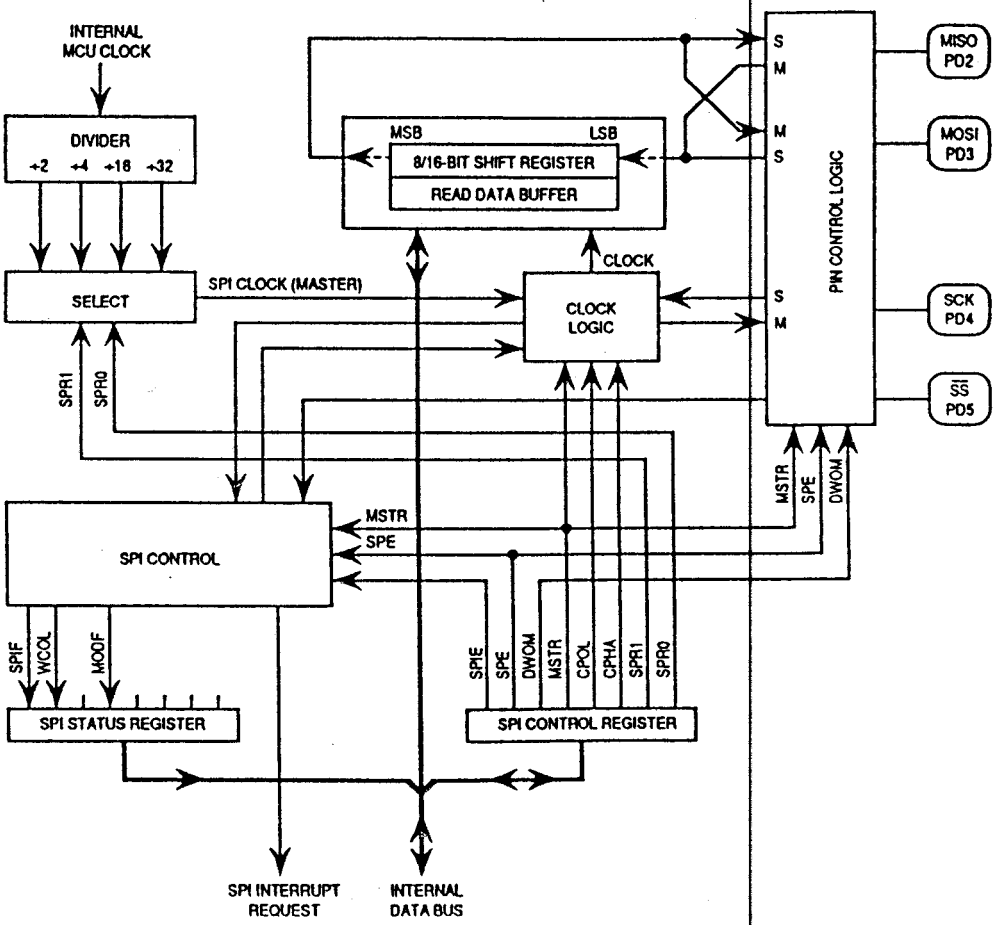
Gambar 2.11 menunjukkan blok diagram dari rangkaian internal Serial Peripheral Interface. Saat peralatan Master mengirimkan data ke peralatan Slave melalui jalur MOSI, peralatan Slave merespon dengan mengirimkan data ke peralatan Master melalui jalur MISO dari peralatan Master. Hal ini merupakan transmisi fullduplex dengan kedua data masuk dan keluar disinkronkan dengan menggunakan sinyal clock yang sama.

Byte yang ditransmit digantikan oleh byte yang diterima dan tidak memerlukan bit data status *transmit-empty* dan *receiver full* yang terpisah. Bit status SPIF digunakan untuk menandai selesainya operasi I/O.

SPI memerlukan buffer ganda pada saat pembacaan, tetapi tidak pada saat penulisan. Jika penulisan dilakukan pada saat transfer data, maka transfer akan tetap berlangsung

tanpa terputus dan penulisan akan gagal. Kondisi ini akan mengakibatkan status bit *Write-Collision* (WCOL) dalam SPSR adalah satu. Setelah data bit digeser, flag SPIF pada SPSR akan satu.

Pada mode Master, pin SCK adalah output. Pin tersebut akan tetap pada level tinggi atau rendah, tergantung pada



Gambar 2.11[13]

Blok Diagram Rangkaian Internal SPI

13 ibid, p. 8-2

bit CPOL dalam SPCR. hingga data ditulis ke shift register yang ada. Pada saat itu delapan clock dihasilkan untuk menggeser delapan bit data dan kemudian SCK konstan pada salah satu level lagi.

II.3.8 Sistem Timer

Sistem timer dari mikrokontroler MC68HC11E9 tersusun dari 5 clock devider. Clock devider utama merupakan counter free running 16 bit. Semua sistem timer direferensikan pada counter ini. Counter ini akan menghitung dari 0000 Hex pada saat reset, sampai FFFF Hex kemudian kembali lagi ke 0000 Hex dan seterusnya, dan tidak diijinkan bagi program pemakai untuk merubah, atau mengintrupsi hitungan ini. Frekuensi dari counter ini ditentukan oleh bit PR[1:0] pada register TMSK2. Bit ini hanya dapat dideklarasikan sekali dan harus dideklarasikan pada 64 cycle pertama sesudah reset. Pada saat reset bit ini bernilai [0:0]. Tabel 2.5 berikut menjelaskan periode dari counter untuk setiap variasi bit PR.

A. Input Capture

Input Capture berfungsi untuk merekam saat dari suatu kejadian eksternal yaitu dengan mencatat angka dari free running counter pada saat terjadi trigger pada pin yang bersesuaian. Angka yang dicatat ini akan disimpan pada register 16 bit TICx sampai ada trigger lagi maka angka ini akan terhapus. Mikrokontroler ini memiliki 3 atau 4 input

capture yang bekerja secara independen antara satu dengan yang lain.

Tabel 2.5[14]

Periode Timer

Control Bits	XTAL Frequencies			
	4.0 MHz	8.0 MHz	12.0 MHz	Other Rates
	1.0 MHz	2.0 MHz	3.0 MHz	(E)
	1000 ns	500 ns	333 ns	(1/E)
PR[1:0]	Main Timer Count Rates			
0 0				
1 count —	1.0 µs	500 ns	333 ns	(E/1)
overflow —	65.536 ms	32.768 ms	21.845 ms	($E/2^{16}$)
0 1				
1 count —	4.0 µs	2.0 µs	1.333 µs	(E/4)
overflow —	262.14 ms	131.07 ms	87.381 ms	($E/2^{18}$)
1 0				
1 count —	8.0 µs	4.0 µs	2.667 µs	(E/8)
overflow —	524.29 ms	262.14 ms	174.76 ms	($E/2^{19}$)
1 1				
1 count —	16.0 µs	8.0 µs	5.333 µs	(E/16)
overflow —	1.049 s	524.29 ms	349.52 ms	($E/2^{20}$)

Sebagai kontrol dari input capture ini dapat dipilih pada bit EDGxB dan EDGxA dari register TCTL2 seperti tabel berikut.

B. Output Compare

Output Compare berfungsi untuk memprogram suatu kegiatan pada waktu tertentu yaitu setiap kali free-running counter mencapai angka yang disimpan pada register TOCx. Untuk setiap dari lima output compare (OC) terdapat compare register 16 bit dan komparator 16 bit. Pada saat angka pada free-running counter sama dengan angka pada compare register

14 ibid, p. 9-3

maka bit yang bersesuaian pada register TFLG1 akan set.

Tabel 2.6[15]

Konfigurasi Kontrol Timer

TCTL2 — Timer Control 2 **$1021**

	Bit 7	6	5	4	3	2	1	Bit 0
	EDG4B	EDG4A	EDG1B	EDG1A	EDG2B	EDG2A	EDG3B	EDG3A
RESET:	0	0	0	0	0	0	0	0

EDGxB	EDGxA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any edge

Tabel 2.7[16]

Periode Real Time Interup

PACTL — Pulse Accumulator Control **$1026**

	Bit 7	6	5	4	3	2	1	Bit 0
	DDRA7	PAEN	PAMOD	PEDGE	DDRA3	I4/O5	RTR1	RTR0
RESET:	0	0	0	0	0	0	0	0

RTR[1:0]	E = 1 MHz	E = 2 MHz	E = 3 MHz	E = X MHz
0 0	2.731 ms	4.096 ms	8.192 ms	$(E/2^{13})$
0 1	5.461 ms	8.192 ms	16.384 ms	$(E/2^{14})$
1 0	10.923 ms	16.384 ms	32.768 ms	$(E/2^{15})$
1 1	21.845 ms	32.768 ms	65.536 ms	$(E/2^{16})$

OC1 memiliki kekhususan jika dibandingkan dengan output

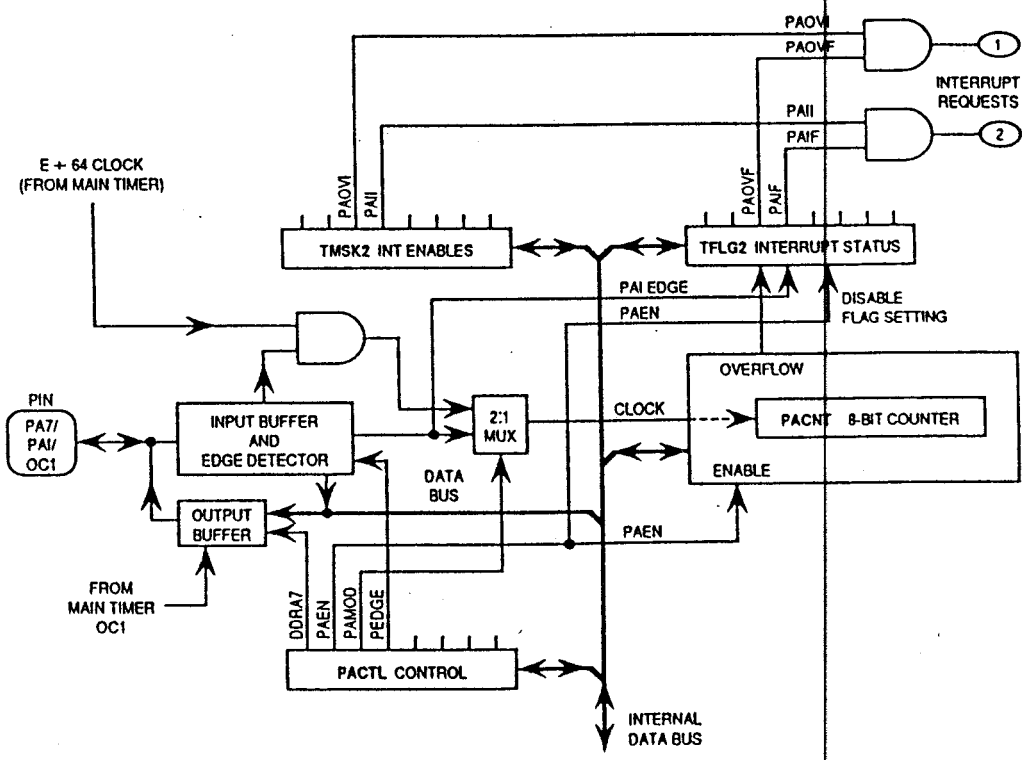
15 ibid, p. 9-6

16 ibid, p. 9-4

compare yang lain yaitu OC1 ini dapat menghasilkan output pada beberapa atau semua dari 5 pin OC. Ini tergantung pada mask register dari OC1 (OC1M), dan data register dari OC1 (OC1D).

C. Real Time Interup

Real Time Interup digunakan untuk membangkitkan interup hardware dengan periode yang tetap. RTI ini dikontrol oleh



Gambar 2.12[17]

Diagram Blok Akumulator Pulsa

[17] ibid, p. 9-19

bit RTR1 dan RTR0 pada register kontrol pulsa akumulator (PACTL). Sumber clock dari RTI ini adalah free-running counter. Frekuensi dari RTI pada setiap frekuensi kristal adalah seperti pada tabel berikut.

D. Akumulator Pulsa

Mikrokontroler MC68HC11E9 memiliki couter 8 bit yang dapat dioperasikan sebagai simple event counter atau gated time accumulation, yang tergantung pada bit PAMOD dari register PACTL.

Pada mode even counting, 8 bit counter ini menerima clock dari pin eksternal dengan clock rate maksimal sebesar setengah dari E clock. Pada mode gated time accumulation, 8 bit counter ini menerima clock dari 1/64 E clock pada saat pin eksternal PAi dalam keadaan aktif. Diagram Blok dari pulsa akumulator adalah seperti pada gambar berikut.

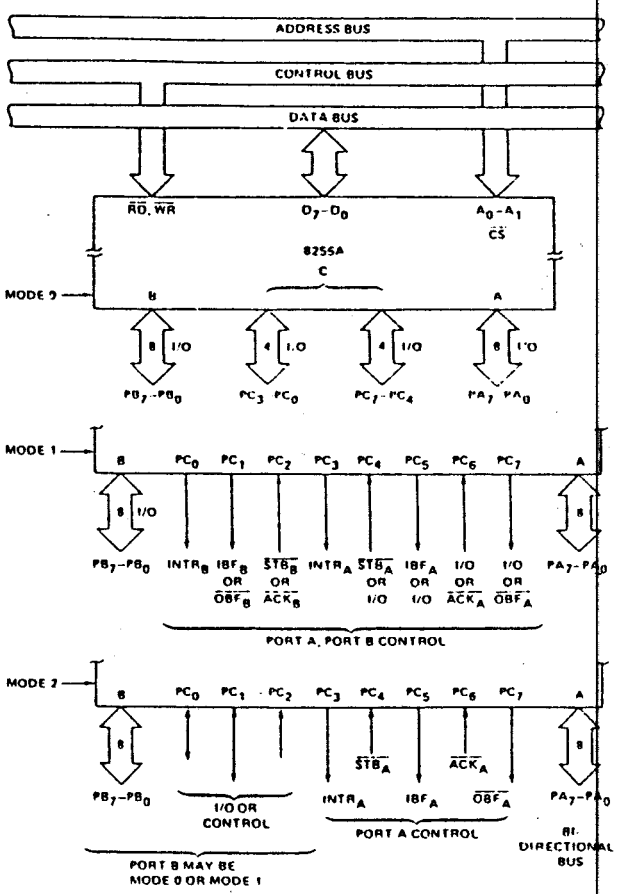
II.4 PPI (PROGRAMMABLE PERIPHERAL INTERFACE) 8255

PPI 8255 adalah peralatan *general purpose programmable I/O* yang didesain untuk digunakan dengan mikroprosesor. PPI 8255 ini mempunyai 24 pin I/O yang dapat diprogram secara individual ke dalam dua group dan dioperasikan ke dalam tiga mode, yaitu mode 0, mode 1, dan mode 2. Penjelasan mengenai ketiga mode tersebut dipaparkan pada bagian berikut ini.

II.4.1 Mode 0 (Basic I/O)

Konfigurasi operasi ini menyediakan operasi-operasi

sederhana untuk input dan output bagi ketiga buah port yang ada. Tidak ada sinyal handshaking yang bisa diberikan ataupun diterima melainkan data secara sederhana dikirim dan dibaca dari port.



Gambar 2.13[18]

Ringkasan Mode Operasi PPI 8255

---

18 Hall, Douglas V., Microprocessors and Interfacing Programming and Hardware, Mc. Graw Hill Book Co., Singapore, 1986, p. 264

II.4.2 Mode 1 (Strobed I/O)

Konfigurasi operasi ini menyediakan fasilitas untuk mentransfer data I/O dari dan ke port tertentu dengan dilengkapi oleh sinyal handshaking. Port A dan port B dapat digunakan untuk transfer data, sedangkan port C sebagai pembangkit sinyal handshaking.

II.4.3 Mode 2 (Strobed Bidirectional I/O)

Konfigurasi operasi ini menyediakan fasilitas untuk komunikasi data 8 bit dua arah dengan peralatan luar. Tersedia sinyal-sinyal untuk handshaking dan interrupt dengan fungsi enable dan disable-nya.

II.5 KOMUNIKASI SERIAL ASINKRON

RS-232C adalah interface elektrik standard untuk menghubungkan komponen-komponen sistem seperti modem, printer, dan komputer. Standard ini pada tahun 1969 ditetapkan oleh EIA (Electronic Industries Association), sebuah organisasi perdagangan industri, kemudian diperluas melalui CCITT (Consultative Comitte on International Telegraphy and Telephony), salah satu komite dari ITU (International Telecommunication Union) yang merupakan salah satu lembaga di PBB yang membidangi masalah telekomunikasi dunia.

RS-232C menetapkan 25 jalur sinyal yang membentuk 18 rangkaian dengan jalur yang kembali lewat ground. Standard juga menetapkan range tegangan untuk logika 0 dan logika 1,

level impedansi, waktu naik / jatuh, laju bit maksimum, dan kapasitansi maksimum yang digunakan dalam semua rangkaian.

Tabel 2.8 menunjukkan nama sinyal, arah sinyal, dan uraian fungsi masing-masing pin RS-232C. Pin-1 Protective Ground secara tak resmi disebut *Chasis Ground*. Jika suatu peralatan tidak mempunyai Ground pada plug AC power supply-nya, maka peralatan tersebut perlu untuk dihubungkan ke pin-1, ini untuk mencegah kejutan listrik.

Fungsi pin ini seringkali dirancukan dengan pin-7 Common Return. Cabang tengah cord AC pada akhirnya harus kembali ke pentanahan induk, oleh karenanya diistilahkan Ground. Dalam prakteknya secara tidak langsung jalur ini juga dapat menghasilkan resistansi antara peralatan dan bumi.

Saat dua peralatan dihubungkan ke jaringan distribusi yang berbeda, maka jalur pembumian keduanya bisa berbeda. Secara elektronik hal tersebut akan menyebabkan casis keduanya tidak sama, walaupun kondisi ini dapat mengganggu komunikasi, tetapi masih dapat diatasi dengan menghubungkan kedua casis lewat pin-1.

Beberapa pin lain yang digunakan dalam interface mikro komputer sebagian besar hanya menggunakan sembilan pin. Sekumpulan pin 2,3,4,5,6,7,8,20 yang sering dipakai disebut Delapan Besar.

Adapter komunikasi serial secara penuh dapat diprogram dan hanya mendukung komunikasi asinkron. Adapter juga akan menyisipkan dan menghapus start-bits, stop-bits, dan

parity-bits.

Generator Baud Rate terprogram memungkinkan beroperasi dari 50 baud sampai 9600 baud. Lima, enam, tujuh, atau delapan bit karakter dengan 1, 1.5, atau 2 stop-bit telah disediakan.

Servis 14h/0 memberikan dua harga output yang menunjukkan status dan kondisi adapter komunikasi asinkron, dimana informasi tersebut tersimpan pada register AH untuk status line dan pada register AL untuk status modem. demikian juga untuk servis 3.

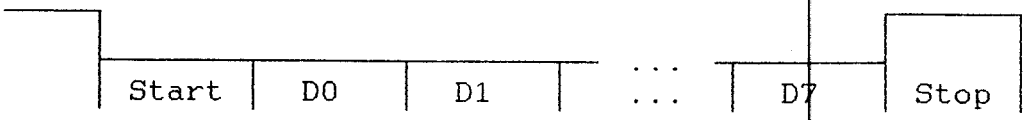
Tabel 2.8[19]

Nama Pin dan Uraian Sinyal RS-232C

PIN NUMBER	COMMON NAME	RS-232-C NAME	DESCRIPTION	SIGNAL DIRECTION ON DCE
1		AA	PROTECTIVE GROUND	–
2	TXD	BA	TRANSMITTED DATA	IN
3	RXD	BB	RECEIVED DATA	OUT
4	$\overline{\text{RTS}}$	CA	REQUEST TO SEND	IN
5	$\overline{\text{CTS}}$	CB	CLEAR TO SEND	OUT
6	$\overline{\text{DSR}}$	CC	DATA SET READY	OUT
7	GND	AB	SIGNAL GROUND (COMMON RETURN)	–
8	$\overline{\text{CD}}$	CF	RECEIVED LINE SIGNAL DETECTOR	OUT
9		–	(RESERVED FOR DATA SET TESTING)	–
10		–	(RESERVED FOR DATA SET TESTING)	–
11			UNASSIGNED	–
12		SCF	SECONDARY REC'D. LINE SIG. DETECTOR	OUT
13		SCB	SECONDARY CLEAR TO SEND	OUT
14		SBA	SECONDARY TRANSMITTED DATA	IN
15		DB	TRANSMISSION SIGNAL ELEMENT TIMING (DCE SOURCE)	OUT
16		SBB	SECONDARY RECEIVED DATA	OUT
17		DD	RECEIVER SIGNAL ELEMENT TIMING (DCE SOURCE)	OUT
18			UNASSIGNED	–
19		SCA	SECONDARY REQUEST TO SEND	IN
20	$\overline{\text{DTR}}$	CD	DATA TERMINAL READY	IN
21		CG	SIGNAL QUALITY DETECTOR	OUT
22		CE	RING INDICATOR	OUT
23		CH/CI	DATA SIGNAL RATE SELECTOR (DTE/DCE SOURCE)	IN/OUT
24		DA	TRANSMIT SIGNAL ELEMENT TIMING (DTE SOURCE)	IN
25			UNASSIGNED	–

19 ibid, p. 451

Data serial dengan format start-bit, 8 bit data, no parity bit, dan 1 stop-bit ditunjukkan pada gambar 2.14. Sedangkan arti tiap bit dari status line dan status modem ditunjukkan pada tabel 2.11.



Gambar 2.14

Format Data Serial

Tabel 2.9[20]

Alamat Adapter Komunikasi

(buku

I/O Decode (in Hex)			
Primary Adapter	Alternate Adapter	Register Selected	DLAB State
3F8	2F8	TX Buffer	DLAB=0 (Write)
3F8	2F8	RX Buffer	DLAB=0 (Read)
3F8	2F8	Divisor Latch LSB	DLAB=1
3F9	2F9	Divisor Latch MSB	DLAB=1
3F9	2F9	Interrupt Enable Register	
3FA	3FA	Interrupt Identification Registers	
3F8	2F8	Line Control Register	
3FC	2FC	Modem Control Register	
3FD	2FD	Line Status Register	
3FE	2FE	Modem Status Register	

I/O Decodes

Hex Address 3F8 to 3FF and 2F8 to 2FF											
A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	DLAB	Register
1	1 0	1	1	1	1	1	x	x	x		
							0	0	0	0	Receive Buffer (read), Transmit Holding Reg (write)
							0	0	1	0	Interrupt Enable
							0	1	0	x	Interrupt Identification
							0	1	1	x	Line Control
							1	0	0	x	Modem Control
							1	0	1	x	Line Status
							1	1	0	x	Modem Status
							1	1	1	x	None
							0	0	0	1	Divisor Latch (LSB)
							0	0	1	1	Divisor Latch (MSB)

Note: Bit 8 will be logical 1 for the adapter designated as primary or a logical 0 for the adapter designated as alternate (as defined by the address jumper module on the adapter)

A2, A1 and A0 bits are "don't cares" and are used to select the different register of the communications chip.

20 . . . . . , Technical Reference for PC/XT System, IBM, USA, 1987, p. 187

Tabel 2.10

Arti Tiap Bit dari Register AL

pada Servis 14 H/0

bit 76543210	Arti
000	110 baud
001	150 baud
010	300 baud
011	600 baud
100	1200 baud
101	2400 baud
110	4800 baud
111	9600 baud
00	No parity
01	Odd parity
10	No parity
11	Even parity
0	1 stop bit
1	2 stop bits
10	7-bit data length
11	8-bit data length

Tabel 2.11

Arti Tiap Bit dari Status Line Register

bit 76543210	Arti
1	Time-out error
1	TSR empty
1	THR empty
1	Break interrupt detected
1	Framing error
1	Parity error
1	Overrun error
1	Data ready

BAB III

PERENCANAAN PERANGKAT KERAS

III.1 PENDAHULUAN

Pada bab ini akan dibahas tentang perencanaan perangkat keras dari peralatan pembaca huruf Braille, dengan menggunakan mikrokontroler 68HC11.

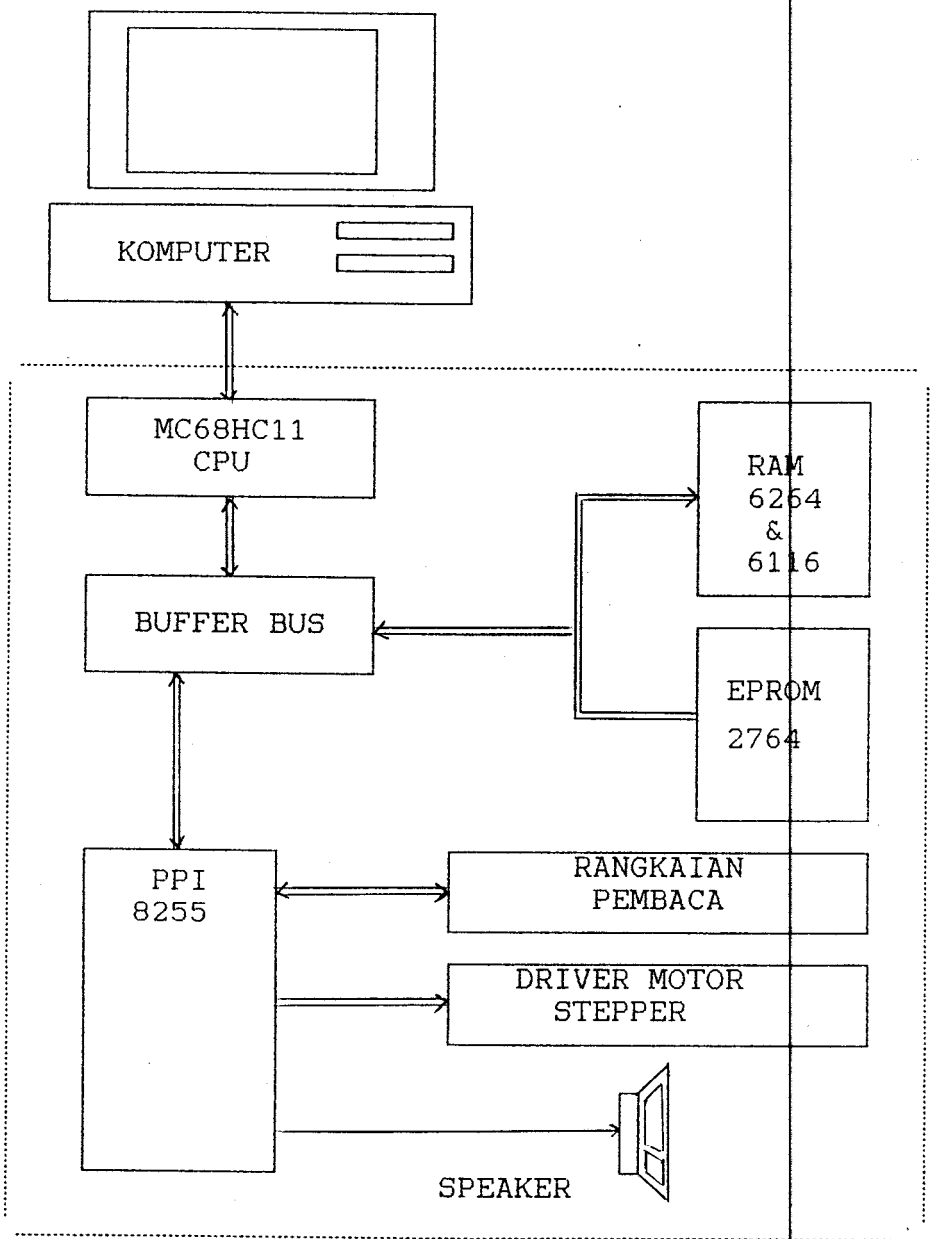
Perencanaan perangkat keras yang digunakan adalah meliputi mikrokontroler MC68HC11E9 yang dalam hal ini digunakan modul EVBU (Evaluation Board Unit) produk dari Motorola, sistem ekspansi memori unit yang terdiri dari RAM dan EPROM, sistem input/output, perencanaan mekanik, rangkaian driver motor serta rangkaian pembaca relief huruf Braille.

III.2 BLOK DIAGRAM PERANGKAT KERAS

Blok diagram dari perangkat keras yang dibuat adalah seperti terlihat pada gambar 3.1 di bawah.

III.3 KOMPUTER IBM PC

Pada peralatan yang dibuat ini komputer IBM PC digunakan untuk menerima data hasil dari pembacaan peralatan untuk kemudian diolah dengan menggunakan editor teks yang digunakan. Pengiriman data antara mikrokontroler dengan komputer dilakukan dengan komunikasi serial standar RS-232C.



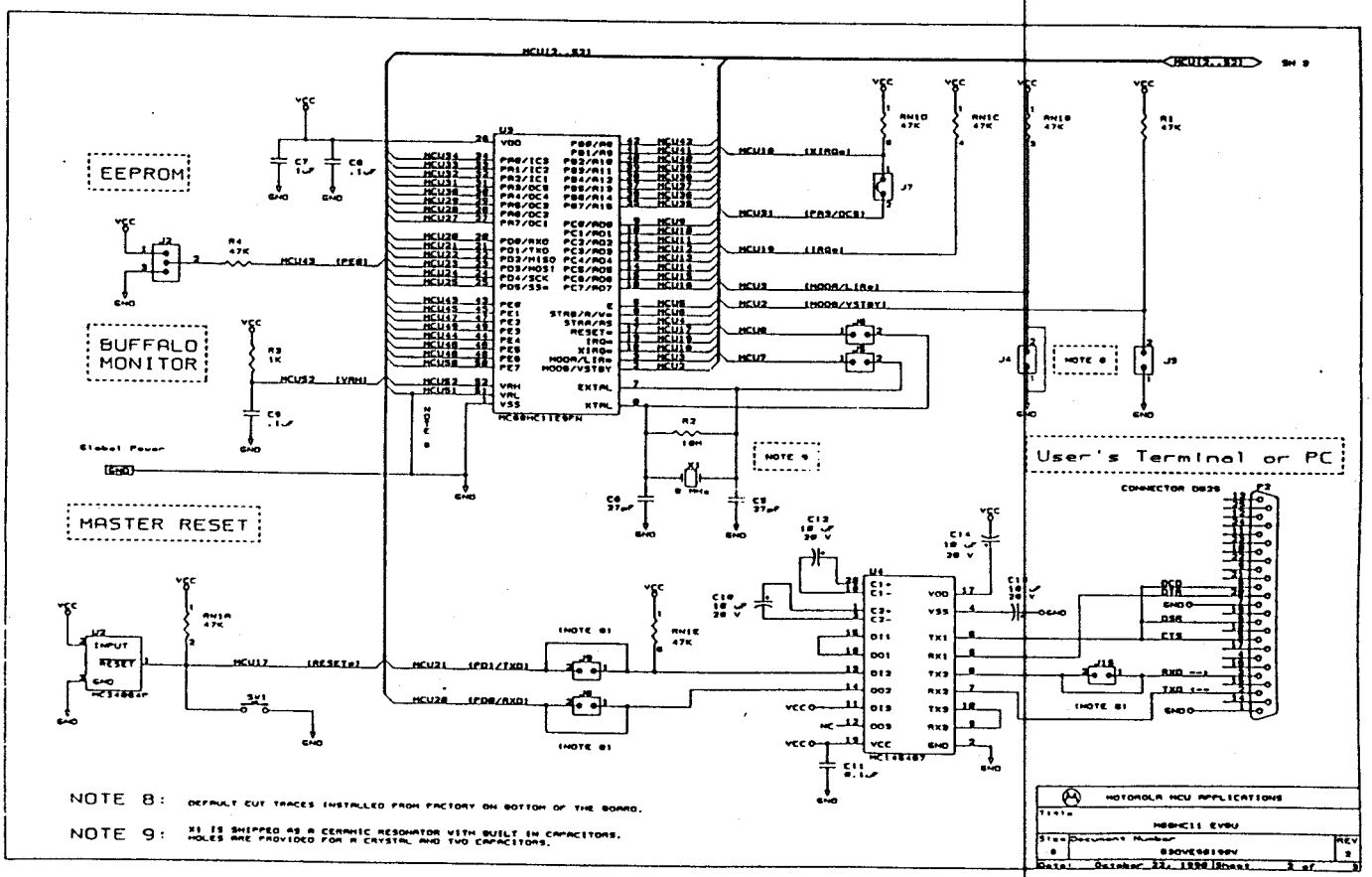
Gambar 3.1

Blok Diagram

III.4 MIKROKONTROLER MC68HC11E9

Pada peralatan yang dibuat, mikrokontroler MC68HC11E9

dioperasikan pada mode expanded multiplexed dimana pin MODA dan MODB diberi level tegangan high. Kristal yang digunakan sebagai pembangkit sinyal clock adalah sebesar 8.0 MHz yang dihubungkan ke pin XTAL dan EXTAL sehingga mikrokontroler bekerja pada frekuensi 2.0 MHz atau seperdelapan dari frekuensi kristal.



Gambar 3.2[21]

Mikrokontroler MC68HC11E9

21 . . . . . , **M68HC11EVBU Evaluation Board User's Manual**, Motorola Inc., USA, 1990, p. 6-15

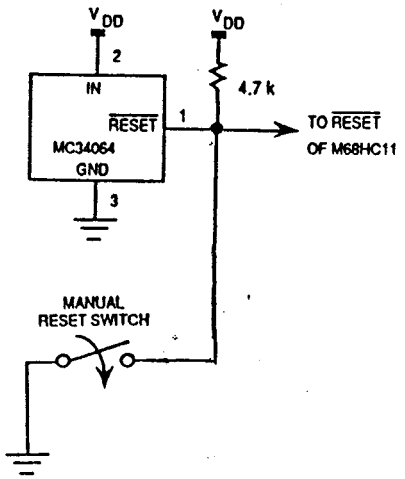
Untuk membangkitkan sinyal RESET digunakan IC MC34064P yang merupakan IC voltage detector dengan pin input dihubungkan ke Vdd sedangkan GND dihubungkan ke ground. Pin RESET dihubungkan ke RESET dari MC68HC11E9 serta resistor pull-up sebesar 47K. Switch RESET menghubungkan pin ini dengan ground sehingga jika tombol reset ditekan akan menghasilkan logika low. Rangkaian selengkapnya dari mikrokontroler MC68HC11E9 adalah seperti terlihat pada gambar 3.2.

III.5 BUFFER

Untuk menghubungkan mikrokontroler MC68HC11E9 dengan rangkaian penunjang lainnya pada peralatan ini, digunakan rangkaian buffer yang berfungsi untuk menambah arus fan-out, serta untuk menjamin bahwa pin input tidak akan berada pada kondisi high impedance yang dapat merusakkan mikrokontroler ini. Selain itu rangkaian buffer ini juga berfungsi untuk memisahkan address dan data low karena pada mode expanded multiplex A0-A7 dan D0-D7 menggunakan bus yang sama.

Berdasarkan timing diagram dari bus pada expanded multiplex mode seperti terlihat pada gambar 3.4, maka untuk bus data digunakan 74LS245 dengan port B dihubungkan ke mikrokontroler sedangkan port A ke rangkaian eksternal. Untuk kontrol DIR digunakan pin R/-W sehingga pada saat high B = A dan sebaliknya, sedangkan untuk pin G dihubungkan ke pin AS (Address Strobe) sehingga pada saat pengiriman address maka 74LS245 ini tidak aktif sedangkan pada saat AS

low data dikirim.



Gambar 3.3[22]

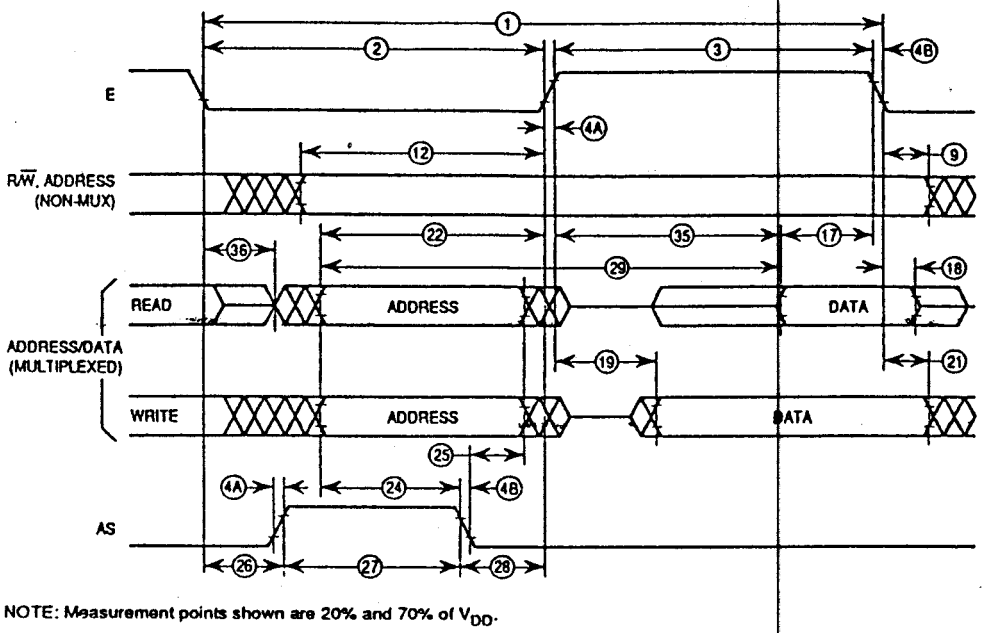
Rangkaian Reset

Untuk address low (A0 - A7) digunakan IC D Flip-Flop 74LS373 dengan kontrol AS sehingga address low ini tetap dapat di-hold sampai adanya address berikut.

Untuk address high (A8 - A15) digunakan 74LS244 dengan semua kontrol dihubungkan ke ground sehingga output selalu mengikuti perubahan input, demikian juga dengan pin AS, E, dan R/-W. Rangkaian selengkapnya dari buffer terlihat pada gambar 3.5.

---

22 MC68HC11E9 Technical Data, op. cit, p. 2-3



Gambar 3.4[23]

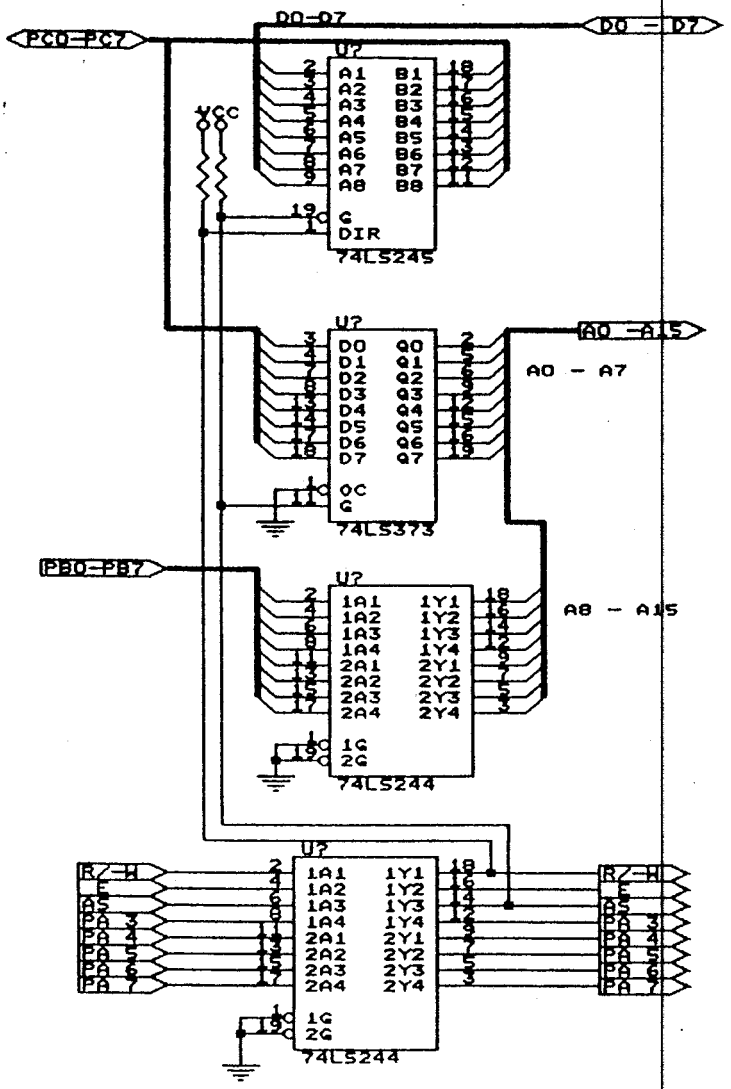
Timing Diagram Mode Expanded Multiplex

Tabel 3.1

Tabel Kebenaran -WE dan -OE

E	R/-W	-WE	-OE
0	0	1	1
0	1	1	1
1	0	0	1
1	1	1	0

[23] ibid, p. A-15



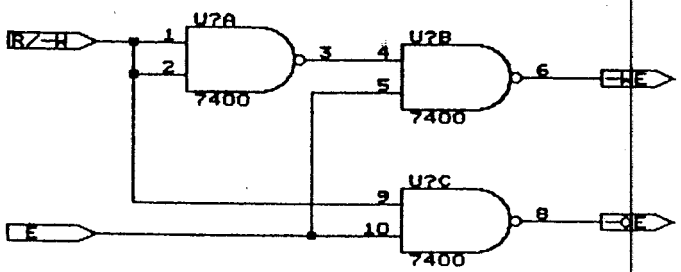
Gambar 3.5

Rangkaian Buffer

Pada bagian ini juga disediakan rangkaian untuk membangkitkan sinyal WRITE (-WE) dan sinyal READ (-OE) sesuai dengan tabel 3.1. Sedangkan rangkaian pembangkit sinyal -WE dan -OE adalah seperti terlihat pada gambar 3.6.

III.6 MEMORI

Peralatan ini membutuhkan memori luar sehingga digunakan expanded multiplex mode. Adapun memori yang digunakan adalah EPROM 2764 (8 KByte), RAM 6264 (8 KByte) serta RAM 6116 (2 KByte). EPROM digunakan sebagai penyimpan program yang akan dieksekusi oleh mikrokontroler sedangkan RAM sebagai tempat pengolahan dan penyimpanan data hasil pembacaan dari head pembaca.



Gambar 3.6

Rangkaian Pembangkit Sinyal -WE dan -OE

Berdasarkan peta memori dari mode expanded multiplex maka memori eksternal ini ditempatkan pada tempat yang kosong sebagai berikut :

- EPROM 2764 : 2000 - 3FFF
- RAM 6264 : 4000 - 5FFF
- RAM 6116 : 6000 - 67FF

Setelah menentukan alamat dari memori maka rangkaian pen-dekode address dibuat berdasarkan tabel pengalamatan memori seperti berikut.

Tabel 3.2

Pengalamatan Memori

MEMORI	ALAMAT	A15	A14	A13	A12
2764	2000-3FFF	0	0	1	x
6264	4000-5FFF	0	1	0	x
6116	6000-67FF	0	1	1	x

x = don't care

Rangkaian dekoder address ini dibuat dengan menggunakan IC 74LS138, rangkaian tersebut dapat dilihat pada gambar 3.7.

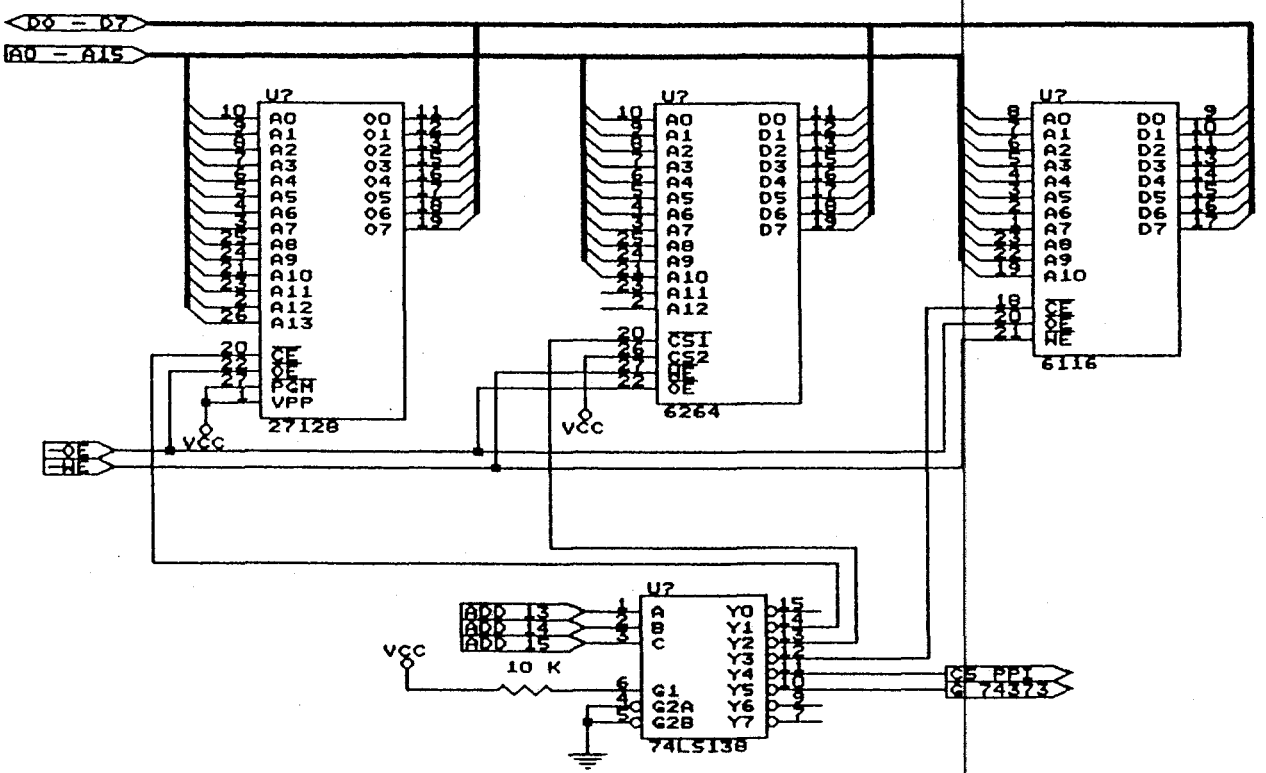
III.7 RANGKAIAN INPUT/OUTPUT

Untuk rangkaian input/output digunakan PPI 8255 serta IC 74LS373 karena dibutuhkan 4 buah port data I/O. PPI 8255 dioperasikan pada mode 0 dengan port A berupa port input yang menerima input dari head pembaca dan kontrol pada peralatan, port B digunakan untuk mengatur gerakan dari motor langkah baik motor ordinat maupun absis. Port C sebagian digunakan sebagai input dari tombol pengontrol sedangkan sebagian lagi sebagai output ke Speaker untuk menghasilkan bunyi. IC 74LS373 digunakan sebagai port output ke LED.

Pengalamatan dari PPI 8255 dan 74LS373 ini adalah sebagai berikut :

- Port A PPI : 8000

- Port B PPI : 8001
- Port C PPI : 8002
- Control Word : 8003
- IC 74LS373 : A000

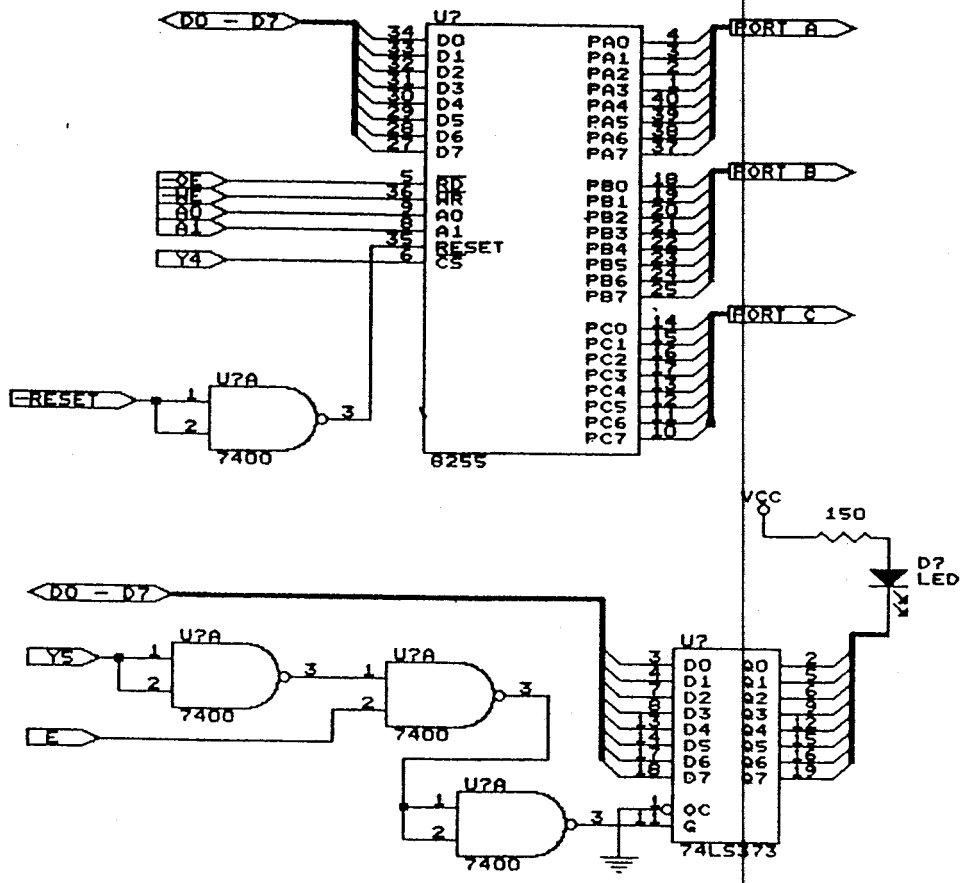


Gambar 3.7

Rangkaian Dekoder Memori Eksternal

Rangkaian input output adalah seperti terlihat pada gambar 3.8. Sebagai dekoder digunakan IC 74LS138 yang sama dengan yang digunakan untuk dekoder memori. Dari rangkaian dekoder yang digunakan terlihat bahwa terjadi bayangan dari alamat PPI 8255 setiap kelipatan empat pada interval

8000-9FFF, namun tidak akan mengganggu fungsi rangkaian karena daerah itu adalah daerah kosong.



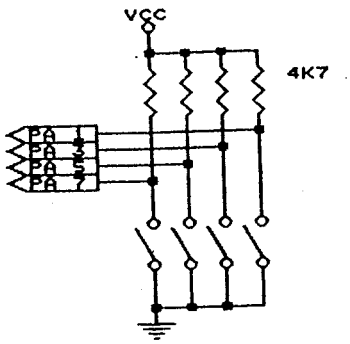
Gambar 3.8

Rangkaian Input Output

III.8 RANGKAIAN PEMBACA

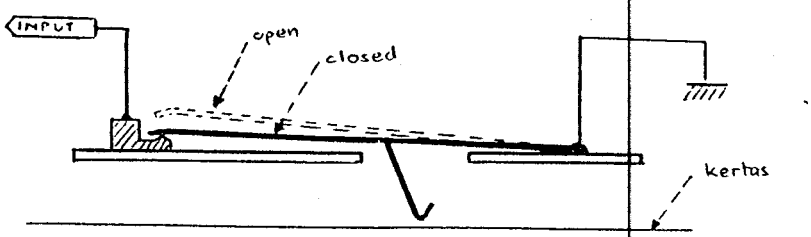
Rangkaian pembaca dihubungkan ke port A dari PPI. Pembaca ini berupa 4 buah pin yang berfungsi sebagai switch normally-closed seperti terlihat pada gambar 3.9. Masing - masing pin dihubungkan ke pin PA1, PA3, PA5, PA7.

Sistem mekanik dari switch yang dibuat adalah dengan menggunakan batang besi selebar ± 2.5 mm. Pada saat menyentuh relief dari huruf Braille batang besi akan terangkat sehingga menghasilkan logika tinggi.



Gambar 3.9

Rangkaian Pembaca



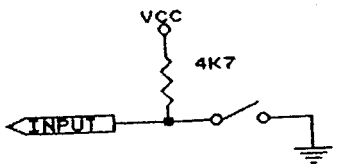
Gambar 3.10

Pin Head Pembaca

III.9 RANGKAIAN KONTROL

Sebagai pengontrol kerja dari peralatan digunakan 4 buah tombol pada key-pad yang masing-masing berfungsi

sebagai ON-LINE, MODE, LINE-FEED serta FORM-FEED. Juga terdapat 2 buah mikroswitch yang berfungsi untuk memantau apakah head sudah berada pada sisi kiri atau kanan sehingga motor dihentikan agar tidak terjadi kerusakan. Rangkaian dari keempat tombol dan dua mikroswitch adalah seperti gambar 3.11. Switch ini dalam keadaan normally open, sehingga dalam keadaan tidak aktif yang terbaca adalah logika high, sedangkan pada saat switch ini ditekan maka akan terhubung ke ground dan terbaca logika low. Resistor yang digunakan untuk pull-up adalah sebesar 4.7 KOhm.

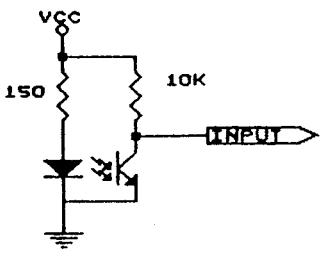


Gambar 3.11

Switch Kontrol

Untuk kontrol awal dan akhir pembacaan serta status kertas digunakan 2 buah foto transistor, seperti dapat dilihat pada gambar 3.12. Foto transistor ini memiliki bidang frekuensi pada daerah infra merah sehingga digunakan LED infra merah untuk menyinari foto transistor. Bila sinar infra merah terkena pada foto transistor akan terjadi bias pada basis sehingga tegangan output akan mendekati ground.

sedangkan sebaliknya jika sinar infra merah terhalang maka tegangan output sama dengan tegangan saturasi.



Gambar 3.12

Foto Transistor

III.10 DRIVER MOTOR STEPPER

Pada peralatan ini digunakan 2 buah motor stepper, masing-masing untuk gerakan searah sumbu X dan searah sumbu Y. Kedua motor stepper merupakan motor stepper empat fasa[24] yang memiliki resolusi sebesar 1.8 derajad/langkah.

Motor stepper X dihubungkan ke port B (PB4 - PB7) sedangkan motor Y dihubungkan ke PB0 - PB3. Untuk membangkitkan arus listrik yang cukup untuk setiap fasa dari motor stepper digunakan transistor daya TIP 122.

Kolektor dari transistor ini dihubungkan ke tegangan suplai 12 Volt dan 15 Volt, sedangkan emiter dihubungkan ke tiap fasa dari motor stepper. Basis dari transistor TIP 122 ini dihubungkan ke port B melalui optokopler. Penggunaan

---

[24] Hall, Douglas V, op. cit, p. 303

optokopler adalah untuk memisahkan secara fisik antara rangkaian elektronik yang memiliki tegangan 5 Volt dan yang lebih dari 5 Volt agar jika terjadi kekeliruan pada rangkaian motor, tidak akan mengakibatkan kerusakan pada mikrokontroler. Untuk membangkitkan arus bias pada basis maka basis dari transistor ini dihubungkan dengan tahanan ke Vdd. Perencanaan besarnya tahanan yang diberikan untuk bias basis adalah sebagai berikut :

β = Ic / Ib

Ib = (Vcc-V_{BE}) / R

maka R = β(Vcc-V_{BE})/Ic

untuk β = 500

Vcc = 15 V

Ic = 0.34 A

R = $\simeq$ 10 KOhm

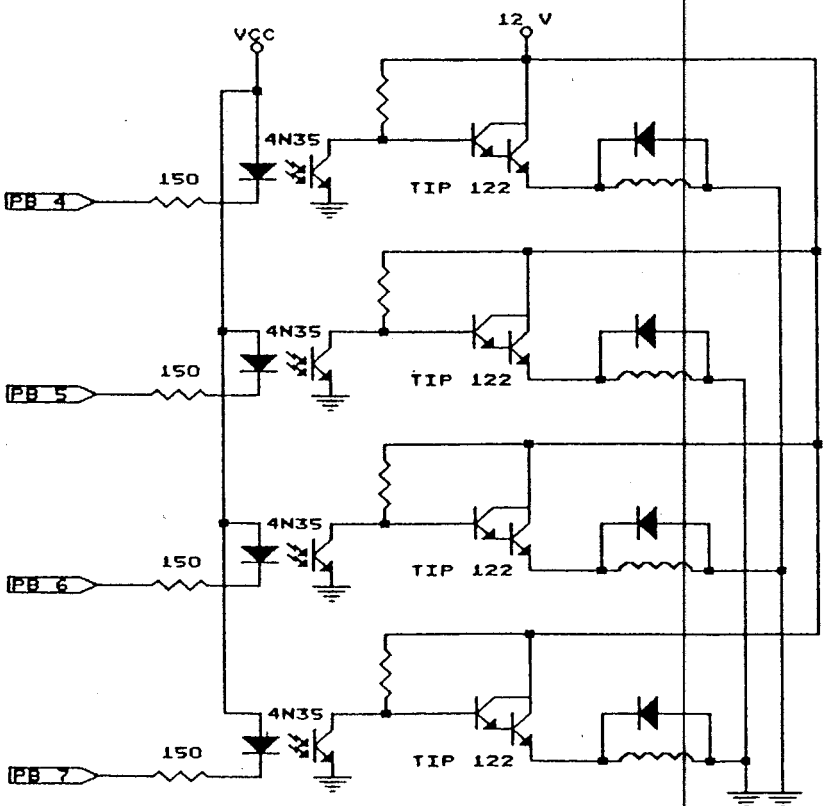
Pada saat output dari port B high maka LED pada optokopler tidak menyala sehingga tegangan output optokopler dalam keadaan cut-off. Kejadian ini akan menyebabkan basis dari TIP 122 terbias, maka akan disuplai arus ke motor stepper. Sebaliknya jika port B menghasilkan logika low maka LED akan menyala sehingga Vbe dari TIP 122 sangat rendah. TIP 122 akan berada dalam keadaan cut-off dan tidak ada arus yang disuplai ke kumparan motor stepper.

Untuk menghasilkan gerakan memutar pada motor stepper maka keempat fasa tersebut harus diberi arus secara bergantian seperti pada tabel 3.3.

Tabel 3.3

Urutan Langkah Motor Y

Step	PB0	PB1	PB2	PB3
1	0	0	1	1
2	0	1	1	0
3	1	1	0	0
4	1	0	0	1

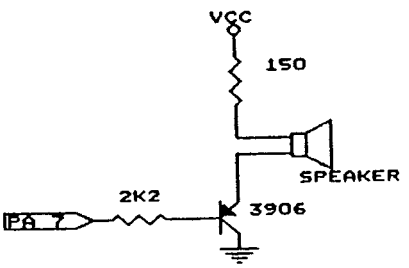


Gambar 3.13

Rangkaian Driver Motor Stepper

III.11 PEMBANGKIT BUNYI

Untuk membangkitkan bunyi morse digunakan timer dari mikrokontroler yang membangkitkan sinyal gelombang kotak dengan frekuensi 600 Hz. Untuk memperkuat sinyal ini digunakan rangkaian buffer seperti pada gambar 3.14.



Gambar 3.14

Rungkaian Buffer Speaker

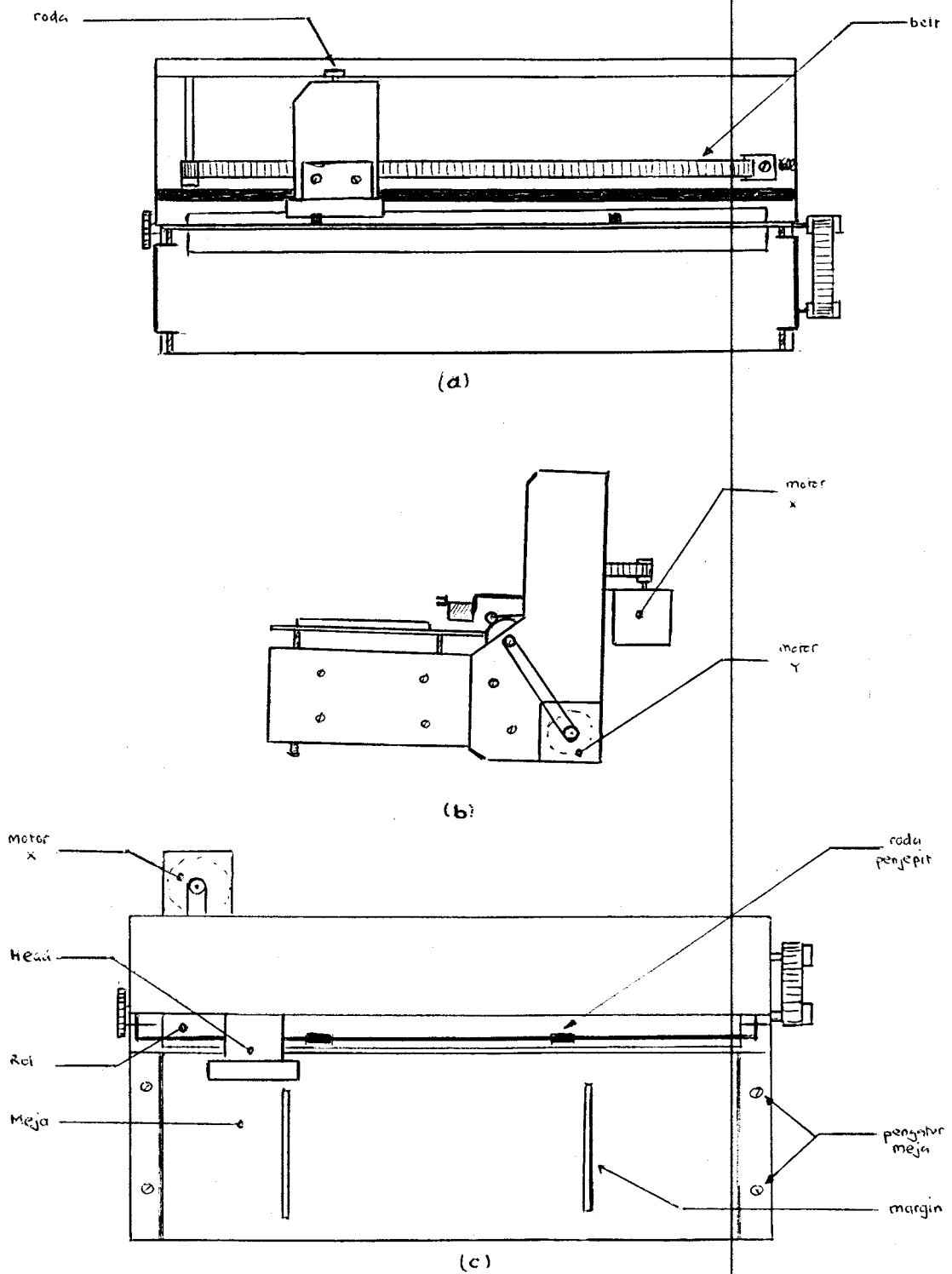
III.12 SISTEM MEKANIK

Sketsa dari sistem mekanik dari peralatan ini adalah seperti terlihat pada gambar 3.15. Untuk menggerakkan head pembaca searah sumbu X digunakan motor stepper. Gerakan dari motor stepper ini ditransferkan ke head melalui belt bergerigi dan pully. Penggunaan belt yang bergerigi ini dimaksudkan agar mengurangi terjadinya slip yang akan merusak hasil pembacaan. Head pembaca diikatkan pada belt sehingga pada saat motor berputar maka head dapat digerakkan ke kiri atau kanan. Sebagai rel tempat head bergerak digunakan batang besi dan roda.

Untuk menggerakkan kertas searah sumbu Y, maka kertas

dijepit oleh beberapa roda kecil pada rol. Rol ini diputar oleh motor stepper Y dengan bantuan belt bergerigi.

Meja kertas terbuat dari kaca yang dapat dinaikkan dan diturunkan untuk mengatur jarak head ke kertas. Pada meja ini juga terdapat pembatas margin kiri dan kanan yang dilengkapi dengan roda-roda kecil. Roda-roda ini berfungsi untuk menjaga agar kertas benar-benar berada dalam posisi yang datar dan tidak terjadi slip pada saat kertas ditarik oleh rol.



Gambar 3.15

Sistem Mekanik

a) tampak depan
b) tampak samping kanan
c) tampak atas

BAB IV

PERENCANAAN PERANGKAT LUNAK

IV.1 PENDAHULUAN

Perangkat lunak yang dibuat adalah berupa program dengan bahasa Cross Assembler MC68HC11 yang kemudian di-compile dengan ASMHC11.EXE. Hasil dari compiler ini adalah berupa S-record atau program dengan ekstensi S19. Program S-record ini sebelum ditulis ke EPROM harus diubah menjadi program yang berisi op-code murni. Untuk ini diperlukan HEXBIN2.EXE yang akan menghasilkan file program dengan ekstensi BIN.

Perangkat lunak untuk komputer IBM PC digunakan bahasa Assembly 8088 yang dikompilasi dengan Turbo Assembler TASMB.

IV.2 DEKLARASI ALAMAT DAN VARIABEL

Pada bagian awal dari program utama pada mikrokontroler MC68HC11E9 ini terdapat pernyataan untuk mendeklarasikan alamat dan variabel yang digunakan pada program ini. Untuk mendeklarasikan alamat dan variabel ini digunakan perintah EQU, FCC, FCB, dan FDB. EQU adalah untuk menyamakan suatu label dengan angka tertentu misalnya :

MODE EQU $4000 artinya MODE = 4000 Hex

FCC adalah untuk mengisi variabel tertentu dengan karakter, FCB untuk mengisi alamat tertentu dengan data byte sedangkan FDB dengan data word (double byte).

Pada bagian ini terdapat deklarasi dari alamat port

pada PPI, port untuk LED, serta alamat dari variabel yang diletakkan pada daerah RAM alamat 67E0 Hex - 67FF Hex. Sebagai tempat penyimpanan data hasil baca dari head pembaca dan tempat pengolahan data dideklarasikan pada daerah RAM 4000 Hex - 5A29 Hex.

Selanjutnya dari bagian ini adalah data-data baik data ASCII, data kode huruf Braille, kode Morse, maupun data mode list dari operasi alat ini.

Pada perangkat lunak ini register kontrol diletakkan pada alamat 1000 Hex - 103F Hex dengan jalan membiarkan bit REG[3:0] dari register INIT berisi logika 0. Deklarasi alamat register-register disesuaikan dengan letak dari register tersebut seperti pada tabel 4.1.

IV.3 INISIALISASI

Program utama ini diawali dengan menginisialisasi sistem. Stack pointer diinisialisasikan pada alamat 6400 Hex, yang menunjuk ke alamat terbawah dari stack dengan perintah :

```
LDS #$6400
```

PPI 8255 diberi control word 91 Hex yaitu untuk mengoperasikan pada mode 0 dengan port A sebagai input, port B sebagai output dan port C lower input serta port C upper untuk output dengan perintah :

```
LDAA #$91
STAA portCW
```

Selanjutnya program ini akan memadamkan semua LED,

speaker, memberikan harga awal pada motor stepper yaitu 00110011 Bin, menunjuk ke mode awal, awal dari lokasi PAGE

Tabel 4.1[25]

Kontrol Register

INIT — RAM and I/O Mapping Register **$103D**

	Bit 7	6	5	4	3	2	1	Bit 0
	RAM3	RAM2	RAM1	RAM0	REG3	REG2	REG1	REG0
RESET:	0	0	0	0	0	0	0	1

REG[3:0]	Address
0000	$0000–$003F
0001	$1000–$103F
0010	$2000–$203F
0011	$3000–$303F
0100	$4000–$403F
0101	$5000–$503F
0110	$6000–$603F
0111	$7000–$703F
1000	$8000–$803F
1001	$9000–$903F
1010	$A000–$A03F
1011	$B000–$B03F
1100	$C000–$C03F
1101	$D000–$D03F
1110	$E000–$E03F
1111	$F000–$F03F

serta harga awal dari variabel-variabel. Pada bagian akhir dilakukan inisialisasi pada SCI (Serial Communications Interface) dengan menggunakan prosedur ONSCI sebagai

25 ibid, p. 4-11

berikut :

```
*****************************
*     inisialisasi SCI      *
*****************************
     onsci      psha
                ldaa #$30
                staa baud
                ldaa #$00
                staa sccr1
                ldaa #$0c
                staa sccr2
                pula
                rts
```

SCI dioperasikan pada baud rate 9600, 1 start dan stop bit, panjang data adalah 8 bit serta tanpa parity.

Pada saat reset maka program akan dimulai pada vektor reset yaitu FFFE, sedangkan lokasi ini adalah lokasi dari ROM yang pada mikrokontroler MC68HC11E9 berisi program BUFFALO (Bit User's Fast Friendly Aid to Logical Operation).

Untuk mengarahkan mikrokontroler ke program yang dibuat yang terletak pada EPROM 2000 Hex dapat ditempuh dengan dua cara. Cara yang pertama adalah dengan mendisable ROM BUFFALO dengan jalan memberikan logika low pada bit ROMON dari register CONFIG. Cara yang kedua adalah dengan memberikan logika high pada pin 0 dari PORT E, karena pada awal dari program BUFFALO pin ini akan dideteksi dan jika high maka program melompat ke EEPROM yaitu pada alamat B600 Hex. Pada EEPROM inilah terdapat perintah untuk melompat ke alamat program utama, yaitu JMP #$2000. Peralatan ini menggunakan cara yang kedua karena dengan cara ini maka program BUFFALO pada ROM masih dapat digunakan dengan sedikit perubahan.

IV.4 PROGRAM UTAMA

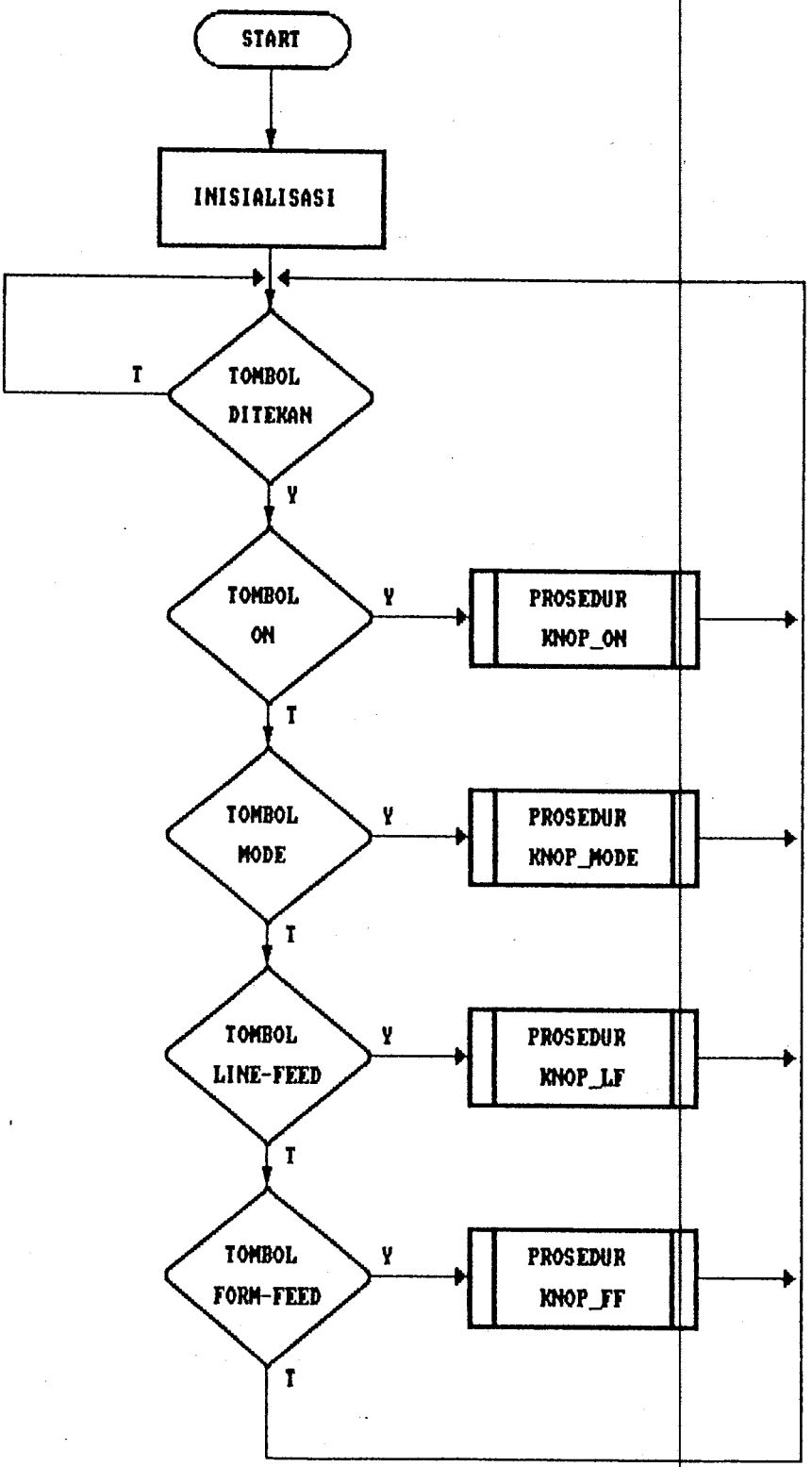
Program utama dari peralatan ini diletakkan pada EPROM eksternal dimulai pada alamat 2000 Hex. Diagram alir dari program utama adalah seperti pada gambar 4.1.

Program ini akan menunggu sampai ada tombol kontrol yang ditekan. Setelah ada tombol yang ditekan maka tombol itu akan dikenali dan program utama akan memanggil prosedur yang sesuai dengan tombol yang ditekan.

Prosedur KNOP_ON adalah untuk memulai proses yang diinginkan sesuai dengan mode yang dipilih. Prosedur KNOP_MODE berfungsi untuk menentukan mode yang dipilih. Mode dari peralatan ini 6 macam, yaitu :

- Baca : membaca tulisan Braille dan menyimpan ke memori.
- Auto : membaca tulisan Braille dan langsung dikirim ke komputer setelah selesai pembacaan satu baris.
- Send : mengirim data yang ada dalam memori ke komputer.
- Send dan Morse : sama dengan send, tetapi juga diubah dalam bentuk bunyi morse
- Rec dan Morse : menerima data dari komputer dan mengubah menjadi bunyi kode morse.
- Auto dan Morse : sama dengan auto tetapi juga diubah menjadi bunyi kode morse.

Pemilihan mode ini dilakukan secara sekuensial setiap kali penekanan tombol MODE yaitu dengan melakukan increment



Gambar 4.1

Diagram Alir Program Utama

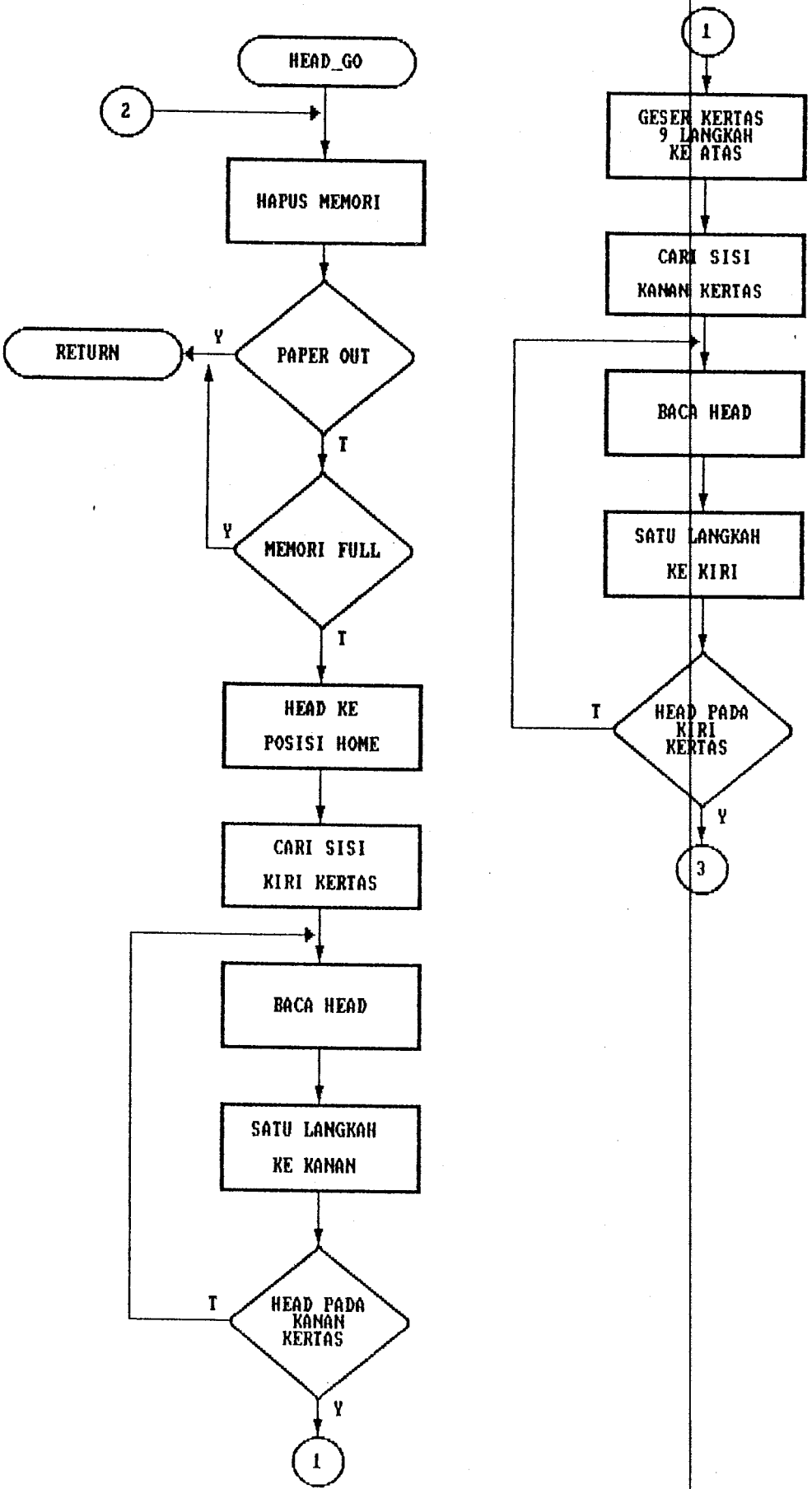
terhadap variabel MODE_PTR.

Prosedur KNOP_LF dan KNOP_FF adalah untuk menggerakkan kertas ke atas. Perbedaan dari keduanya adalah LF (Line Feed) menggerakkan sejauh satu baris, sedangkan FF (Form Feed) sejauh satu halaman. Kedua prosedur ini akan mendeteksi apakah head telah berada pada posisi paling kiri. Jika belum maka head akan digeser ke kiri sampai switch batas kiri tertekan oleh head, baru gerakan ke atas dilakukan. Hal ini adalah untuk menghindari kerusakan pin head pada saat bergesekan dengan kertas.

IV.5 PEMBACAAN HURUF BRAILLE

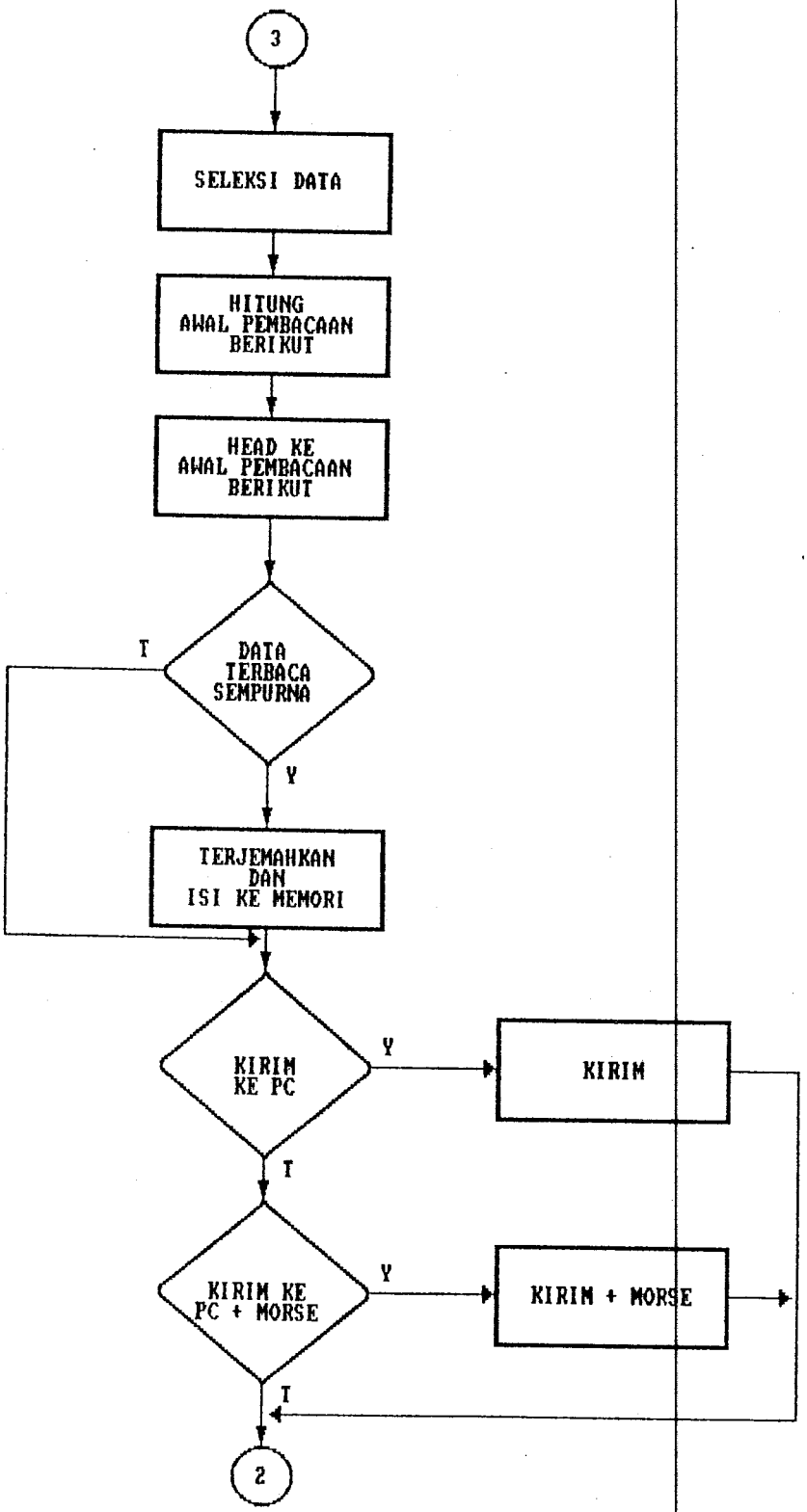
Untuk menangani pembacaan huruf pada kertas digunakan prosedur HEAD_GO. Tugas dari prosedur ini adalah mengatur gerakan dari motor dan melakukan pembacaan dari head pembaca. Setelah pembacaan satu baris selesai maka data tersebut akan dipisahkan antara data yang sebenarnya dengan noise yang terbaca. Prinsip dari seleksi ini adalah data yang hanya muncul sekali dianggap merupakan noise.

Setelah data tersebut diseleksi maka akan diperiksa apakah pembacaan tadi telah sempurna. Jika tidak sempurna misalnya hanya sisi sebelah atas yang terbaca, maka prosedur ini akan menggeser head ke awal pembacaan yang benar dan melakukan lagi pembacaan. Jika pembacaan berhasil maka data terbaca tersebut diterjemahkan ke dalam bentuk ASCII dan disimpan ke lokasi memori yang kapasitasnya 2000 karakter. Jika lokasi memori ini telah penuh maka pembacaan tidak akan



Gambar 4.2

Diagram Alir Prosedur HEAD_GO



Gambar 4.2 (lanjutan)

Diagram Alir Prosedur HEAD_GO

dilakukan sampai lokasi memori ini dikosongkan dengan mengirim ke komputer atau reset. Diagram alur dari prosedur HEAD_GO adalah seperti ditunjukkan dalam gambar 4.2 dan prosedur ini selengkapnya dapat dilihat pada lampiran.

Setelah pembacaan ini, jika mode operasi adalah AUTO maka karakter yang ada dalam memori tersebut akan langsung dikirim.

IV.6 MENGIRIM DATA KE KOMPUTER

Pengiriman data ke komputer dilakukan oleh prosedur SEND. Data karakter yang dikirim ke komputer adalah sesuai dengan yang terdapat dalam lokasi memori 2000 byte. Lokasi memori ini disusun dalam bentuk lingkaran, penunjuk awal dan akhir memori adalah variabel PAGE_HEAD dan PAGE_TAIL. Jika PAGE_HEAD sama dengan PAGE_TAIL maka memori dalam keadaan penuh sedangkan jika PAGE_HEAD tepat di depan PAGE_TAIL maka memori dalam keadaam kosong.

Sebelum data dikirim maka dilakukan pengecekan apakah data register dalam keadaan kosong yaitu dengan membaca SCSR. Bit 7 dari register SCSR ini akan set jika SCDR dalam keadaan kosong yaitu jika komputer telah membaca data kiriman. Jika sudah kosong maka data yang akan dikirim diletakkan ke register SCDR. Cuplikan dari program mengirim data adalah sebagai berikut :

```
*********************************
*     Kirim 1 Byte ke SCI       *
*                               *
*     kiriman di SCIBUF         *
*     error   di SCIWARN        *
*********************************

kirim       psha
            pshb
            pshx
            pshy

            ldx #$0fff

            ldy #scsr
kirim1      brset 0,y,#%10000000,kirim2
            dex
            bne kirim1

            ldaa #$ff
            staa sciwarn
            bra kirim3

kirim2      ldaa scibuf
            staa scdr
            ldaa #0
            staa sciwarn

kirim3      puly
            pulx
            pulb
            pula
            rts
```

Dalam prosedur ini jika dalam beberapa saat ternyata kiriman tidak diambil oleh komputer maka akan ada pesan error karena dianggap komputer dalam keadaan tidak siap untuk menerima data.

IV.7 MENERIMA DATA DARI KOMPUTER

Penerimaan data dari komputer dilakukan oleh prosedur TERIMA. Untuk mengetahui apakah terdapat kiriman dari komputer pada jalur RS-232 maka program akan memeriksa bit 1

(FE) dari register SCSR, jika bit ini set berarti data yang dikirim telah berada pada register SCDR. Selanjutnya data yang diinginkan dapat diambil dari register SCDR yang terletak di alamat $102F. Cuplikan dari prosedur terima adalah sebagai berikut :

```
*********************************
*     Terima 1 Byte ke SCI      *
*                               *
*     output di SCIBUF          *
*********************************

terima      psha
            pshb
            pshx
            pshy

            ldy #scsr
terima1     brset 0,y,#%00000010,terima2
            jsr sw_press
            ldaa sw_flag
            cmpa #$ff
            bne terima1

            ldaa #$ff
            staa scibuf
            bra terima3

terima2     ldaa scdr
            staa scibuf

terima3     puly
            pulx
            pulb
            pula
            rts
```

Data karakter yang diterima dalam bentuk kode ASCII ini akan diubah ke dalam bentuk bunyi kode morse oleh prosedur RX_MR. Fungsi dari menerima data ini adalah untuk menghasilkan kode morse dari tulisan di layar komputer.

IV.8 PENGATUR GERAKAN MOTOR STEPPER

Terdapat 4 buah prosedur yang digunakan untuk mengatur gerakan dari motor stepper yaitu KIRI, KANAN, ATAS dan prosedur BAWAH. Prinsip algoritma dari keempat prosedur ini sama, kecuali pada prosedur KIRI dan KANAN yang mengatur gerakan dari head pembaca selalu dilakukan pengecekan apakah head telah berada pada posisi maksimum kiri atau kanan. Jika telah berada pada posisi maksimum maka motor tidak dipaksa bergerak lagi.

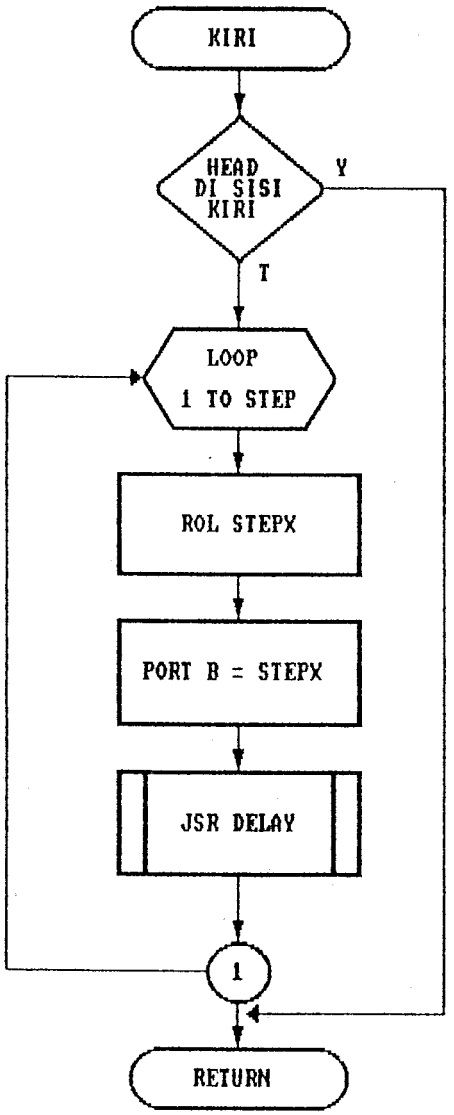
Diagram alir dari prosedur KIRI adalah seperti gambar di bawah ini. Untuk delay digunakan prosedur DLY10 yang melakukan delay selama 10 mdet sebagai berikut :

```
*************************************************
*  Prosedur DLY10 untuk delay selama 01 * 10 ms  *
*************************************************

dly10       pshx
            ldx #1
dly         pshx
            ldx #$0d06
dloop       dex
            bne dloop
            pulx
            dex
            bne dly

            pulx
            rts
```

Prinsip utama dari 4 prosedur ini adalah rotasi data sedangkan pada instruction set dari assembler MC68HC11 tidak terdapat perintah rotasi, melainkan hanya perintah shift yaitu LSL dan LSR. Perintah ini menggeser data 1 bit ke kanan atau kiri, dimana bit yang kosong akan diisi dengan 0



Gambar 4.3

Diagram Alir Prosedur KIRI

dan bit yang terletak di ujung arah pergeseran akan diisikan ke CARRY. Untuk membangkitkan rotasi dari perintah geser ini digunakan program yang selalu memeriksa carry, jika set maka data yang baru dimasukkan diubah menjadi 1 seperti cuplikan program berikut :

```
            lsr stepX
            bcc kiri_2
            ldaa stepX
            adda #$80
            staa stepX
kiri_2      ldaa stepX
            anda #$0f
            staa portB
```

IV.9 PEMBANGKIT BUNYI

Untuk membangkitkan bunyi dengan frekuensi tertentu maka speaker harus diberi sinyal yang dalam hal ini adalah sinyal kotak dengan frekuensi yang tetap.

Pembangkitan sinyal kotak ini dengan memanfaatkan sistem timer pada MC68HC11 yaitu dengan menggunakan output compare 1 (OC1), karena hanya output compare 1 yang responnya dapat langsung dioutputkan ke pin output.

Sistem timer MC68HC11 tidak menyediakan mode khusus untuk menghasilkan astable multifibrator, sehingga untuk membangkitkan sinyal kotak dengan durasi tertentu dilakukan dengan program. Pertama-tama program menulis logika high ke register OC1D dan menuliskan angka yang mewakili lama waktu sampai output menjadi high ke register TOC1. Setelah free running counter mencapai harga ini maka pin output akan high, sehingga program harus menulis logika 0 ke register

OC1D dan waktu durasi ke register TOC1. Jika proses ini dilakukan berulang-ulang maka dihasilkan sinyal kotak pada pin OC1.

Untuk menghasilkan bunyi kode morse digunakan prosedur MORSE dan BUNYI. Sinyal yang dihasilkan adalah dengan frekuensi 600 Hz. Untuk menentukan angka counter pada saat high dan low dilakukan perhitungan sebagai berikut :

- Untuk bit PR[1:0] = 0,0 maka 1 hitungan 500 ns
- T = 1/f = 1/600 = 1.666 ms

Jumlah hitungan 1 periode = 1.666 ms / 500 ns = 3333

Jadi hitungan saat high dan low adalah 1/2 (3333) = 1666

Cuplikan prosedur pembangkit sinyal kotak adalah sebagai berikut :

```
                ldaa #$80       ; inisilisasi output compare 1
                staa oc1m
bunyi1          ldaa #$80       ; data high
                staa oc1d

                ldd toc1        ; inisialisasi TOC1
                addd #1666
                std toc1
bunyi2          ldx #tflg1      ; tunggu OC1F set
                brclr 0,x,#%10000000,bunyi2
                bclr 0,x,#%01111111

                ldaa #$0
                staa oc1d

                ldd toc1
                addd #1666
                std toc1
bunyi3          ldx #tflg1
                brclr 0,x,#%10000000,bunyi3
                bclr 0,x,#%01111111

                bra bunyi1
```

IV.10 PROGRAM PADA IBM PC

Program pada komputr IBM PC adalah berupa program untuk menerima dan mengirim data melalui komunikasi serial ke mikrokontroler MC68HC11. Program ini diletakkan secara residen pada memori komputer IBM PC sehingga dapat bekerja dalam semua pengolah kata yang digunakan.

Untuk menerima data dari mikrokontroler MC68HC11 digunakan prosedur interupsi yang diletakkan pada vektor interupsi 1CH yaitu interupsi tick timer. Interupsi ini akan diakses dengan frekuensi 18.2 Hz setiap kali komputer melakukan penghitungan waktu. Prosedur ini memeriksa apakah ada permintaan kiriman dari mikrokontroler, jika tidak ada maka program selesai, tetapi jika ada maka data kiriman tersebut akan diambil dan akan diletakkan pada BUFFER KEYBOARD sehingga data tersebut akan dianggap sama dengan data yang dari keyboard komputer. Komunikasi serial dilakukan pada baud rate 9600 dan semua proses komunikasi serial dengan menggunakan INT 14H fungsi 0,1,2, dan 3.

Pengisian data pada BUFFER KEYBOARD dilakukan dengan memanfaatkan pointer HEAD dan TAIL, dan jika sudah penuh maka komunikasi dihentikan untuk sementara seperti cuplikan program berikut :

```
mov ax,0040h
mov es,ax
mov bx,[es:0082h]        ;buffer_end
mov buff_end,bx
mov bx,[es:0080h]        ;buffer_start
mov buff_start,bx
mov bx,[es:001ah]        ;buffer_head
mov buff_head,bx
mov bx,[es:001ch]        ;buffer tail
mov si,bx
inc bx
```

```
                inc bx
                cmp bx,buff_end
                jne res_1
                mov bx,buff_start

res_1:          cmp bx,buff_head        ;apakah BUFFER FULL
                je res_no

                mov ah,2                ;ambil kiriman
                mov dx,0
                int 14h
                and ah,10000000b
                jnz res_no              ;komunikasi error

                xor ah,ah
                mov [es:si],ax
                mov [es:001ch],bx
```

Untuk pengiriman data ke mikrokontroler digunakan prosedur interupsi yang diletakkan pada vektor interupsi 09h yaitu keyboard interrupt, sehingga setiap kali tombol keyboard ditekan maka karakter yang ditekan akan dikirim ke mikrokontroler dan akan dihasilkan kode morse dari karakter tersebut.

BAB V

PENGUJIAN DAN PENGUKURAN

V.1 PENDAHULUAN

Setelah peralatan ini selesai dibuat maka dilakukan pengujian dan pengukuran unjuk kerja dari peralatan ini. Tujuan dari melakukan pengujian dan pengukuran adalah untuk mengetahui seberapa jauh kehandalan dari alat ini.

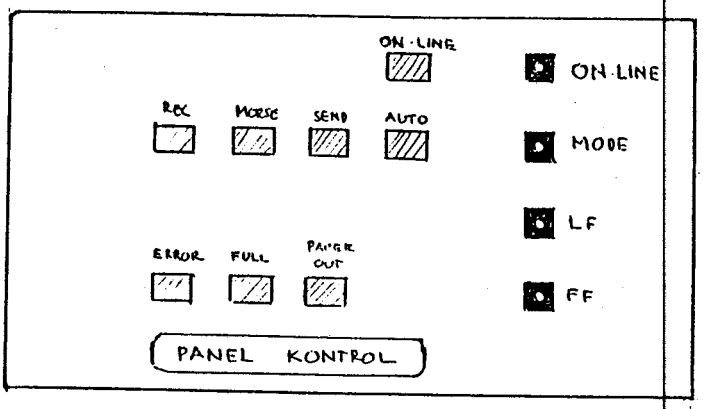
Pada bab ini akan dibahas mengenai cara pemakaian alat, pengukuran data-data antara lain output dari E clock, output dari pembangkit bunyi, output dari head pembaca. Juga akan dilakukan analisa data untuk mengetahui apakah data yang diperoleh sudah sesuai dengan yang direncanakan.

V.2 PEMAKAIAN ALAT

Alat yang dibuat dapat dipakai dalam 6 mode operasi. Pemilihan operasi dan pengontrolan dapat dilakukan melalui tombol kontrol yang terdiri dari 4 buah tombol, masing-masing :

- ON-LINE : untuk memulai/menghentikan opereasi
- MODE : untuk memilih mode operasi
- LINE-FEED (LF): untuk menggerakkan kertas ke atas sejauh 2 mm
- FORM-FEED (FF): untuk mengerakkan kertas ke atas sejauh 20 cm

Alat ini dapat bekerja setelah switch ON/OFF ditekan ke arah ON. Jika lampu pada switch ini menyala berarti alat



Gambar 5.1

Panel Kontrol

telah dalam keadaan siap, jika tidak menyala kemungkinan terjadi kesalahan pada alat.

Penjelasan pemakaian alat dari setiap mode adalah sebagai berikut :

1. Mode BACA

   a. Tombol ON/OFF ditekan ke arah ON.

   b. Kertas dimasukkan ke meja kertas. Jika head berada di antara margin kanan dan kiri sehingga mengganggu pemasukkan kertas, head digeser ke kiri dengan jalan menekan tombol LF sampai head berhenti di sebelah kiri.

   c. Kertas digeser ke atas sampai benar-benar terjepit dengan rol, yaitu dengan menekan tombol LF.

   d. Tombol MODE ditekan sampai semua lampu LED mode tidak menyala, yang menandakan mode BACA.

e. Setelah tombol ON-LINE ditekan maka LED ON-LINE akan menyala dan pembacaan dimulai. Pembacaan ini akan terus berlangsung sampai kertas telah selesai, yang ditandai dengan LED PAPER OUT.

f. Bila sementara proses ini berlangsung diinginkan untuk menginterupsi atau menghentikan proses maka tombol ON-LINE ditekan sampai LED ON-LINE padam.

2. Mode AUTO

Cara pengoperasian mode ini hampir sama dengan mode BACA, hanya terdapat perbedaan yaitu hasil baca tiap baris akan langsung dikomunikasikan ke komputer.

a. Langkah awal dari mode ini sama dengan langkah a, b, dan c pada mode BACA.

b. Tombol MODE ditekan sampai LED AUTO menyala.

c. Mode ini hanya bisa diaktifkan jika hubungan ke komputer sudah terpasang, dan pada komputer telah diaktifkan program residen BRAILE serta teks editor yang digunakan.

d. Langkah selanjutnya sama dengan pada mode BACA.

e. Jika pembacaan berhasil maka tulisan yang dibaca akan langsung tampak pada layar komputer.

3. Mode SEND

Mode ini bertujuan untuk mengirim hasil baca yang telah ada di dalam memori ke komputer. Setelah hasil baca ini dikirim ke komputer maka akan terhapus dari memori alat.

a. Seperti pada mode AUTO, maka hubungan komunikasi serial ke kompiuter harus sudah terpasang, serta program residen BRAILE telah diaktifkan.

b. Tombol MODE ditekan untuk memilih mode SEND yang ditandai dengan LED.

c. Tombol ON-LINE ditekan dan pengiriman data akan terus berlangsung sampai LED ON-LINE padam. Tulisan yang terbaca akan terlihat pada layar komputer.

4. Mode SEND dan MORSE

Mode ini akan sama dengan mode SEND dengan sedikit perbedaan yaitu tulisan yang terdapat dalam memori juga akan dihasilkan dalam bentuk bunyi kode morse.

a. Sama dengan langkah pada mode SEND tetapi tombol mode harus ditekan sampai LED SEND dan MORSE menyala.

b. Akan terdengar bunyi morse dari speaker.

5. Mode AUTO dan MORSE

Mode ini sama dengan mode AUTO tetapi juga ditambah dengan bunyi morse.

a. Langkah pengoperasian mode ini sama dengan mode AUTO dengan perbedaan yaitu tombol mode ditekan sampai LED AUTO dan MORSE menyala.

6. Mode REC dan MORSE

Mode ini bertujuan untuk menghasilkan bunyi kode morse dari karakter yang diketikkan pada keyboard komputer.

a. Mode ini hanya dapat dipergunakan jika komunikasi

serial dengan komputer telah terpasang.

b. Pemilihan mode dilakukan dengan menggunakan tombol MODE dan ditandai dengan LED REC dan MORSE.

c. Tombol ON-LINE ditekan hingga LED akan menyala.

d. Pada komputer diaktifkan program residen MORSE. Jika tombol pada keyboard komputer ditekan maka akan terdengar bunyi kode morse dari tombol yang ditekan. Jika tombol yang ditekan tidak dikenali oleh kode morse maka akan dibatalkan.

Setelah pemakaian alat pastikan bahwa suplai listrik telah dimatikan dengan menekan tombol ON/OFF ke posisi OFF.

V.3 PENGUKURAN SINYAL E CLOCK

Untuk mengetahui frekuensi dari mikrokontroler maka dilakukan pengukuran sinyal E clock dari mikrokontroler. Pengukuran dilakukan dengan menghubungkan pin E ke input Osiloskop sedangkan ground dari osiloskop dihubungkan ke ground dari mikrokontroler.

Data yang diperoleh adalah sebagai berikut :

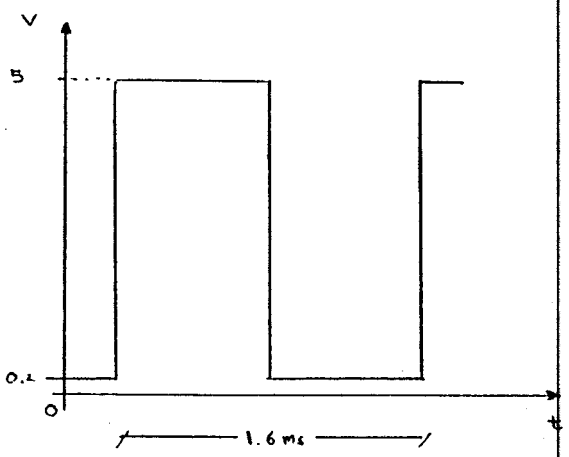
Tabel 5.1

Pengukuran E clock

	Pengukuran	Data Book	Satuan
Lebar pulsa saat high	300	222	ns
Lebar pulsa saat low	200	227	ns
Tegangan saat low	0.2	0.4 (max)	V
Tegangan saat high	5.0	4.8 (min)	V

V.4 PENGUKURAN OUTPUT COMPARE 1

Sinyal output compare 1 ini diperoleh dengan mengukur sinyal yang terdapat pada PA7. Sinyal ini digunakan untuk menghasilkan bunyi pada speaker. Hasil pengukuran adalah seperti terlihat pada gambar 5.2.



Gambar 5.2

Sinyal Output Compare 1

Dari hasil pengukuran diperoleh periode sinyal adalah 1.6 mdet, jadi frekuensi sinyal :

$$f = 1/T = 1/1.6 \text{ mdet} = 625 \text{ Hz}$$

Dalam perencanaan frekuensi sinyal ini adalah 600 Hz. Perbedaan yang terjadi sebesar 25 Hz yang disebabkan oleh pembulatan angka counter disesuaikan dengan kecepatan maksimum free running counter yaitu 500 ns, serta pembacaan

pada layar osiloskop.

V.5 PENGUKURAN KECEPATAN PEMBACAAN

Untuk mengetahui waktu yang dibutuhkan untuk melakukan sekali pembacaan dilakukan pengukuran waktu pembacaan. Pengukuran dilakukan dengan mencatat waktu yang dibutuhkan untuk membaca 1 baris tulisan pada kertas berukuran folio. Hasil pengukuran adalah sebagai berikut :

- Waktu untuk sekali lintasan head : 34 detik
- Waktu untuk pembacaan satu baris tulisan : 2 x 34 = 68 detik.
- Satu lembar kertas dengan 30 baris tulisan adalah : 30 x 68 = 2040 detik atau 34 menit.

Dari hasil pengukuran waktu untuk satu lembar kertas yaitu 34 menit maka kerja dari alat ini masih relatif lambat. Hal ini disebabkan karena jika gerakan head terlalu cepat maka resultan gaya gratitasi dan gaya gerak head akan menggangu hasil pembacaan dari head pembaca. Untuk mempercepat kerja dari alat ini maka diperlukan konstruksi head yang tahan terhadap gaya-gaya ini.

BAB VI
PENUTUP

VI.1 KESIMPULAN

Setelah berbagai pembahasan mengenai alat pembaca huruf Braille yang dibuat, pengukuran serta pengujian yang dilakukan, maka beberapa kesimpulan yang dapat diperoleh adalah sebagai berikut :

Mikrokontroler MC68HC11 adalah mikrokontroler yang menyediakan fasilitas ADC, timer, komunikasi serial sinkronous dan asinkronous, memori RAM serta EEPROM, sehingga sangat tepat penggunaannya untuk alat pembaca huruf Braille yang dibuat ini dan juga alat-alat lainnya yang membutuhkan fasilitas-fasilitas tersebut serta prosesor yang handal.

Mikrokontroler MC68HC11 memiliki empat mode operasi, yaitu mode Single-Chip, Expanded Multiplexed, Special Bootstrap, dan Special Test. Hal ini memudahkan pemakai untuk pemakaian yang membutuhkan I/O port dalam jumlah yang banyak atau membutuhkan memori yang besar. Pada alat pembaca huruf Braille yang dibuat ini digunakan mode Expanded Multiplexed dengan memori eksternal berupa 8 KByte ROM dan 10 KByte RAM yang dapat diakses dengan baik.

Perangkat lunak MC68HC11 yang menggunakan Assembler MC68HC11 merupakan perangkat lunak yang mudah dipelajari dan dipakai karena menggunakan istilah-istilah yang gampang diingat. Fasilitas perangkat lunak BUFFALO versi 3.2 yang

terdapat pada ROM internal dari MC68HC11E9 sangat membantu untuk mempelajari mikrokontroler ini.

Peralatan pembaca huruf Braille yang dibuat ini dapat membantu menerjemahkan naskah dalam bentuk huruf Braille ke dalam naskah latin dan dapat disimpan dalam media penyimpan. Penerjemahan ini akan menolong orang yang tidak tunanetra untuk membaca naskah Braille.

Penggunaan tranduser relief berupa switch sentuh untuk membaca huruf Braille dapat bekerja dengan cukup baik.

Dari hasil pengujian disimpulkan bahwa alat ini dapat bekerja dengan cukup baik. Kesalahan yang terjadi pada alat ini disebabkan oleh kondisi teks Braille, jarak meja pembaca dengan head pembaca serta beberapa kesalahan yang tidak terduga.

VI.2 SARAN

Alat pembaca huruf Braille yang dibuat ini dapat dikembangkan lebih jauh dengan menggunakan head pembaca yang lebih baik, sehingga kehandalan alat dapat ditingkatkan.

Pengembangan alat pembaca huruf Braille ini dapat dilakukan dengan menambah fasilitas pencetak huruf Braille, sehingga alat ini dapat berfungsi seperti mesin fotocopy.

Melalui tugas akhir ini diharapkan dapat meningkatkan perhatian kita terhadap saudara-saudara kita yang menderita cacat dan memberikan wawasan untuk menciptakan alat bantu bagi penderita cacat.

DAFTAR PUSTAKA

1. Hall, Douglas V., **MICROPROCESSORS AND INTERFACING, PROGRAMMING AND HARDWARE**, McGraw Hill Book Co., Singapore, 1986.
2. Peatman, John B., **DESIGN WITH MICROCONTROLLERS**, Prentice Hall International Ltd., USA, 1990.
3. Uffenbeck, John, **THE 8086/8088 FAMILY, DESIGN PROGRAMMING AND INTERFACING**, Prentice Hall International Ltd., USA.
4. ....., **M68HC11EVBU EVALUATION BOARD USER'S MANUAL**, Motorola Inc., USA, 1990.
5. ....., **PEDOMAN PENGGUNAAN HURUF BRAILLE MENURUT EJAAN BAHASA INDONESIA YANG DISEMPURNAKAN**, Depdikbud RI, Jakarta, 1976.
6. ....., **M68HC11 REFERENCE MANUAL**, Motorola Inc., Phoenix Arizona USA, 1991.
7. ....., **MC68HC11E9 TECHNICAL DATA**, Motorola Inc., Phoenix Arizona USA, 1991.
8. ....., **TECHNICAL REFERENCE FOR PC/XT SYSTEM**, IBM, USA, 1987.

```
*****************************************
* Listing Program Mikrokontroler MC68HC11 *
*           PEMBACA HURUF BRAILLE         *
*       DENGAN MIKROKONTROLER MC68HC11    *
*             Nemuel Daniel Pah           *
*               (2882201112)              *
*                                         *
* Last Update: 25 Jan 1994                *
*****************************************

*-------------------------------
* alamat untuk PPI 8255 dan LED
*-------------------------------
portA       equ $8000
portB       equ $8001
portC       equ $8002
portCW      equ $8003

portLED     equ $a000

*-------------------------------
*      alamat variabel
*-------------------------------

page_head   equ $67e0       ; word

page_tail   equ $67e2       ; word

morsebuf    equ $67e4
mr          equ $67e5
shift       equ $67e6
shl         equ $67e7
numlok      equ $67e8
invers      equ $67e9
kode        equ $67ea
kodef       equ $67eb
total1      equ $67ec

knop        equ $67f0
stepX       equ $67f1
stepY       equ $67f2
step        equ $67f3       ; word

simbol      equ $67f5
total       equ $67f6
scibuf      equ $67f7
sciwarn     equ $67f8
mode        equ $67f9
mode_ptr    equ $67fa       ; word

geser       equ $67fc
full        equ $67fd
ascii_reg   equ $67fe
sw_flag     equ $67ff

*-------------------------------
*     alamat buffer data
*-------------------------------
buffer      equ $4000       ; 4000 - 47cf (2000 byte)
```

```
lpf         equ $47e0       ; 47e0 - 4faf (2000 byte)
datanya     equ $4fc0       ; 4fc0 - 514f ( 200 byte)
page        equ $5180       ; 5180 - 594f (2000 byte) ASCII
rxBuf       equ $5960       ; 5960 - 59c4 ( 100 byte)
transit     equ $59c6       ; 59c6 - 5a29 ( 100 byte)

*--------------------------------
*   alamat register-register
*--------------------------------
baud        equ $102b
sccr1       equ $102c
sccr2       equ $102d
scsr        equ $102e
scdr        equ $102f

oc1m        equ $100c
oc1d        equ $100d
toc1        equ $1016
tflg1       equ $1023

*--------------------------------
* TABEL ASCII YANG DIPERLUKAN
*--------------------------------
            org $2b00

ascii       fcc '()'            ; 0-1
            fcc 'ABCDEFGHIJ'    ; 2-11
            fcc 'KLMNOPQRST'    ; 12-21
            fcc 'UVWXYZ'        ; 22-27
            fcc 'abcdefghij'    ; 28-37
            fcc 'klmnopqrst'    ; 38-47
            fcc 'uvwxyz'        ; 48-53
            fcc '1234567890'    ; 54-63
            fcc '..'            ; 64-65
            fcc ',,'            ; 66-67
            fcc ';;'            ; 68-69
            fcc ':::'           ; 70-72
            fcc '??'            ; 73-74
            fcc '!!'            ; 75-76
            fcc '""""'          ; 77-80
            fcc '---'           ; 81-83
            fcc '//'            ; 84-85
            fcc '.+-x.[[[]]]'   ; 86-96
            fcc '{}() '         ; 97-101
            fcc '= + - * '

braile      fcb $33,$33
            fcb $84,$86,$c4,$e4,$a4,$c6,$e6,$a6,$c2,$e2
            fcb $85,$87,$c5,$e5,$a5,$c7,$e7,$a7,$c3,$e3
            fcb $95,$97,$f2,$d5,$f5,$b5
            fcb $04,$06,$44,$64,$24,$46,$66,$26,$42,$62
            fcb $05,$07,$45,$65,$25,$47,$67,$27,$43,$63
            fcb $15,$17,$72,$55,$75,$35
            fcb $0c,$0e,$4c,$6c,$2c,$4e,$6e,$2e,$4a,$6a
            fcb $32,$b2
            fcb $02,$82
            fcb $03,$83
            fcb $22,$a2,$2a
            fcb $13,$93
```

```
                fcb $23,$a3
                fcb $13,$93,$31,$b1
                fcb $11,$91,$19
                fcb $41,$c1
                fcb $0a,$1a,$29,$1c,$58,$37,$b7,$3f,$73,$f3,$7b
                fcb $1b,$39,$5a,$2d,$00

morseLST        fcb $36,$6d
                fcb $05,$18,$1a,$0c,$02,$12,$0e,$10,$04,$17
                fcb $0d,$14,$07,$06,$0f,$16,$1d,$0a,$08,$03
                fcb $09,$11,$0b,$19,$1b,$1c
                fcb $05,$18,$1a,$0c,$02,$12,$0e,$10,$04,$17
                fcb $0d,$14,$07,$06,$0f,$16,$1d,$0a,$08,$03
                fcb $09,$11,$0b,$19,$1b,$1c
                fcb $2f,$27,$23,$21,$20,$30,$38,$3c,$3e,$3f
                fcb $ad,$ad
                fcb $73,$73
                fcb $78,$78
                fcb $78,$78,$78
                fcb $4c,$4c
                fcb $ad,$ad
                fcb $52,$52,$52,$52
                fcb $61,$61,$61
                fcb $29,$29
                fcb $ad,$2a,$61,$19,$ad,$36,$36,$36,$6d,$6d,$6d
                fcb $36,$6d,$36,$6d,$80
                fcb $31,$80,$2a,$80,$19,$80,$19,$80

scan            fcb $6a,$6b
                fcb $7e,$90,$8e,$80,$72,$81,$82,$83,$77,$84
                fcb $85,$86,$92,$91,$78,$79,$70,$73,$7f,$74
                fcb $76,$8f,$71,$8d,$75,$8c
                fcb $1e,$30,$2e,$20,$12,$21,$22,$23,$17,$24
                fcb $25,$26,$32,$31,$18,$19,$10,$13,$1f,$14
                fcb $16,$2f,$11,$2d,$15,$2c
                fcb $02,$03,$04,$05,$06,$07,$08,$09,$0a,$0b
                fcb $34,$34
                fcb $33,$33
                fcb $27,$27
                fcb $87,$87,$87
                fcb $95,$95
                fcb $62,$62
                fcb $88,$88,$88,$88
                fcb $0c,$0c,$0c
                fcb $35,$35
                fcb $43,$6d,$0c,$2d,$34,$1a,$1a,$1a,$1b,$1b,$1b
                fcb $7a,$7b,$6a,$6b,$99
                fcb $0d,$39,$6d,$39,$0c,$39,$69,$39

dftasc          fcb 41,34,15,8,41,52,15,26,28,36,2,10,28,48,2,22
*                               'ngNGnyNYaiAlauAU'
dftbrl          fdb $7600,$f600,$5600,$d600,$1400,$9400,$5200,$d200

dblbrl          fdb $2222,$1212,$2121,$4141

*-------------------------------
*  data standar untuk Proc CARI
*-------------------------------
                org *
```

```
standar     fdb $0204          ; 66dd
            fdb $0608
            fdb $1018
            fdb $2040
            fcb $60
standE      fcb $00

artinya     fdb $0101          ; 66e7
            fdb $0102
            fdb $0202
            fdb $0404
            fcb $04
artiE       fcb $00

*---------------------------
* data untuk mode operasi
*---------------------------
            org *              ; 66f1

mode_lst    fcb %11111111      ; baca
            fcb %11101111      ; auto
            fcb %11110111      ; send
            fcb %11110011      ; send + morse
            fcb %11111001      ; rec  + morse
modeE       fcb %11101011      ; auto + morse


*****************************
* this is the main program  *
*****************************

            org $2000

mulai       lds #$6400

            ldaa #$91          ; control word PPI 8255
            staa portCW        ; PA inp, PB out, PCl inp, PCu out

            ldaa #$00          ; PB dimatikan
            staa portB

            ldaa #$0f          ; SPEAKER off
            staa portC

            ldaa #$ff          ; LED dimatikan
            staa portLED
            staa knop

            ldaa #%00110011    ; inisialisasi MOTOR
            staa stepX
            staa stepY

            ldx #mode_lst      ; inisialisasi MODE & MODE_PTR
            stx mode_ptr
            ldaa 0,x
            staa mode
            staa portLED
```

```
            ldx #page           ; inisialisasi PAGE_TAIL & PAGE_HEAD
            stx page_tail
            inx
            stx page_head

            ldaa #0             ; inisialisasi variabel
            staa shift
            staa shl
            staa numlok
            staa invers
            staa full

            jsr onsci

            ldaa #$0f
            staa morsebuf
            jsr morse

main_1      jsr knop_0

            ldaa knop
            cmpa #$07
            bne main_2
            jsr knop_on

main_2      ldaa knop
            cmpa #$0b
            bne main_3
            jsr knop_mode

main_3      ldaa knop
            cmpa #$0d
            bne main_4
            jsr knop_lf

main_4      ldaa knop
            cmpa #$0e
            bne main_1
            jsr knop_ff

            bra main_1

*********************************
*        prosedur SW_PRESS      *
*    output SW_FLAG = 0  (no)   *
*                   = ff (press)*
*********************************
sw_press    psha
            pshx

            ldaa #$0
            staa sw_flag

            ldx #portC
            brset 0,x,#%00001000,sw_end

            ldx #25
sw_loop     jsr dly10
            dex
```

```
            bne sw_loop

            ldx #portC
            brset 0,x,#%00001000,sw_end

            ldaa #$ff
            staa sw_flag

sw_end      pulx
            pula
            rts

*****************************
*      Prosedur KNOP_0      *
* jika tidak ada input maka *
* tetap dalam prosedur ini  *
* === juga paper out/in === *
*****************************
knop_0      psha
            pshx

noknop      ldx #portA
            brclr 0,x,#%01000000,noknop1
            ldx #mode
            bset 0,x,#%10000000    ; paper-out OFF
            ldaa mode
            staa portLED
            bra noknop2

noknop1     ldx #mode
            bclr 0,x,#%10000000    ; paper-out ON
            ldaa mode
            staa portLED

noknop2     ldaa portC
            anda #$0f
            cmpa #$0f
            beq noknop

press       staa knop
            ldaa portC
            anda #$0f
            cmpa #$0f
            bne press

            ldx #mode
            bset 0,x,#%00100000    ;pesan error
            ldaa mode
            staa portLED

            pulx
            pula
            rts

*****************************
*    prosedure HEAD_GO      *
*****************************

head_go     psha
```

```
            pshb
            pshx
            pshy

head1       ldx #portA              ; head ke kiri
            brclr 0,x,#%00000100,head_awal

            ldx #10
            stx step
            jsr kiri

            jsr sw_press
            ldaa sw_flag
            cmpa #$ff
            beq head0

            bra head1

head_awal   jsr clear_mem

            ldx #portA
            brclr 0,x,#%01000000,head0  ; paper out ?

            ldaa full               ; full ?
            cmpa #$ff
            bne head2
            ldx #mode
            bclr 0,x,#%01000000
            ldaa mode
            staa portLED
head0       jmp head_end

*--------------------
head2       ldx #1                  ; mencari awal pembacaan
            stx step
            jsr kanan
            ldx #portA
            brclr 0,x,#%00010000,head2

head3       ldx #1
            stx step
            jsr kanan
            ldx #portA
            brset 0,x,#%00010000,head3

*--------------------
            ldx #buffer

head_kanan  ldy #1                  ; baca ke kanan
            sty step
            jsr kanan

            jsr sw_press
            ldaa sw_flag
            cmpa #$ff
            beq head1_end

            ldaa portA
```

```
            anda #%10101010
            staa 0,x
            inx
            ldy #portA
            brclr 0,y,#%00010000,head_kanan
*-------------------
            ldy #250                ; gerak ke atas
            sty step
            jsr kanan

            jsr sw_press
            ldaa sw_flag
            cmpa #$ff
            beq head1_end

            ldy #9
            sty step
            jsr atas
*------------------------
            bra head4
head1_end   jmp head_end
*------------------------
head4       ldy #1                  ; mencari awal baca ke kiri
            sty step
            jsr kiri
            ldy #portA
            brclr 0,y,#%00010000,head4

head5       ldy #1
            sty step
            jsr kiri
            ldy #portA
            brset 0,y,#%00010000,head5

*------------------------
head_kiri   ldaa portA              ; baca ke kiri
            anda #%10101010
            lsra
            dex
            oraa 0,x
            staa 0,x

            ldy #1
            sty step
            jsr kiri

            jsr sw_press
            ldaa sw_flag
            cmpa #$ff
            beq head1_end

            ldy #portA
            brclr 0,y,#%00010000,head_kiri

            ldy #250                ; gerakan head ke kiri
            sty step
            jsr kiri
```

```
                jsr filter
                jsr trans

                ldaa total
                staa total1

head7           ldaa #0
                staa geser

                ldaa total
                oraa #%01111110
                cmpa #%01111110
                bne head8

                ldx #8
head7_1         lsl total
                bcs head7_2
                inc geser
                dex
                bne head7_1

head7_2         ldaa geser
                cmpa #2
                bgt head12

                ldaa #8
                staa geser
                bra head12

head8           cmpa #%01111111
                beq head10

                ldx #8
head9           inc geser
                lsl total
                bcc  head10
                dex
                bne head9

                bra head12

head10          ldx #8
head11          lsl total
                bcs head12
                inc geser
                dex
                bne head11

head12          ldaa geser
                cmpa #8
                bgt head13
                cmpa #2
                bgt head14
                cmpa #2
                beq head15

head13          ldaa #8
                staa geser
```

```
head14      ldaa geser
            suba #2
            ldab #5
            mul
            std step
            jsr atas

head15      ldaa geser
            cmpa #8
            bne head5_1

            ldaa total1
            cmpa #0
            beq head5_1

            anda #%10000000
            cmpa #%10000000
            beq head5_1

            jsr to_braile
            jsr to_ascii
            jsr to_page

head5_0     ldaa full                ; full ?
            cmpa #$ff
            bne head5_1
            ldx #mode
            bclr 0,x,#%01000000
            ldaa mode
            staa portLED
            jmp head_end

head5_1     ldaa mode                ; apakah dikirim
            anda #%00011110
            cmpa #%00001110
            bne head6

            ldaa #0
            staa mr
            jsr send
            bra head_15

head6       cmpa #%00001010          ; apakah dikirim + morse
            bne head_15

            ldaa #$ff
            staa mr
            jsr send

head_15     jsr sw_press
            ldaa sw_flag
            cmpa #$ff
            beq head_end

            jmp head_awal

head_end    puly
            pulx
            pulb
```

```
            pula
            rts

*****************************
*    prosedur  KNOP_ON      *
*****************************
knop_on     psha
            pshx

            ldx #mode
            bclr 0,x,#%00000001   ; on-line ON
            ldaa mode
            staa portLED

            ldaa mode
            anda #%00011110

            cmpa #%00010110       ; apakah send ?
            bne on_1

            ldaa #0
            staa mr
            jsr send
            bra on_4

on_1        cmpa #%00010010       ; apakah send + morse ?
            bne on_2

            ldaa #$ff
            staa mr
            jsr send
            bra on_4

on_2        cmpa #%00011000       ; apakah rx   + morse ?
            bne on_3

            jsr rx_mr
            bra on_4

on_3        jsr head_go

on_4        ldx #mode             ; on-line OFF
            bset 0,x,#%00000001
            ldaa mode
            staa portLED

on_5        ldx #portC                      ; sampai ON-LINE
            brclr 0,x,#%00001000,on_5 ; dilepas

            ldx #10
on_6        jsr dly10
            dex
            bne on_6

            pulx
            pula
            rts
```

```
*********************************
*       prosedur KNOP_MODE      *
*                               *
*  untuk menentukan mode operasi *
*  dari peralatan = MODE        *
*********************************

knop_mode   psha
            pshb
            pshx

            ldx mode_ptr
            inx
            cpx #modeE
            ble kn_md1
            ldx #mode_lst

kn_md1      ldaa 0,x
            stx mode_ptr

            ldab mode
            orab #%00011110
            stab mode

            anda mode
            staa mode
            staa portLED

            pulx
            pulb
            pula
            rts

*********************************
*       prosedur KNOP_LF        *
*********************************

knop_lf     psha
            pshx

kn_lf1      ldx #portA
            brclr 0,x,#%00000100,kn_lf2

            ldx #10                 ; head ke kiri
            stx step
            jsr kiri
            bra kn_lf1

kn_lf2      ldx #10
            stx step
            jsr atas

            pulx
            pula
            rts
```

```
*********************************
*       prosedur KNOP_FF        *
*********************************

knop_ff     psha
            pshx

kn_ff1      ldx #portA
            brclr 0,x,#%00000100,kn_ff2

            ldx #10                 ; head ke kiri
            stx step
            jsr kiri
            bra kn_ff1

kn_ff2      ldx #1000
            stx step
            jsr atas

            pulx
            pula
            rts

*********************************
* prosedur RX_MR                *
* menerima kiriman dari PC      *
* output dalam bentuk morse     *
*********************************
countr      rmb 1

rx_mr       psha
            pshb
            pshx
            pshy

rx_0        jsr terima
            ldaa scibuf
            cmpa #$ff
            beq rx_end

            ldx #scan
            ldaa #102
            staa countr
            ldaa scibuf

rx_1        cmpa 0,x
            beq rx_2
            inx
            dec countr
            bne rx_1

            bra rx_0

rx_2        xgdx
            subd #scan
            addd #morseLST
            xgdx
```

```
            ldaa 0,x
            staa morsebuf
            jsr morse
            bra rx_0

rx_end      puly
            pulx
            pulb
            pula
            rts

*********************
* prosedur SEND     *
* MR = 0 (no morse) *
*    = 1 (morse)    *
*********************
send        psha
            pshb
            pshx
            pshy

            ldaa #0
            staa full
            ldx #mode
            bset 0,x,#%01000000
            ldaa mode
            staa portLED

send1       ldx page_tail
            inx
            cpx page_head
            beq send_end

            ldx page_tail
            ldab 0,x
            ldaa #0
            addd #ascii
            xgdy
            ldaa 0,y
            staa scibuf
            jsr kirim

            ldaa sciwarn
            cmpa #$ff
            bne send11

            ldx #mode
            bclr 0,x,#%00100000 ;pesan error
            ldaa mode
            staa portLED
            jmp send_end

send11      ldy #mr
            brclr 0,y,#%00000001,send3
            ldab 0,x
            ldaa #0
            addd #morseLST
            xgdy
            ldaa 0,y
```

```
                staa morsebuf
                jsr morse

send3           jsr sw_press            ; apakah ON-LINE press
                ldaa sw_flag
                cmpa #$ff
                beq send_end

                cpx #$594f
                bge send2
                inx
                stx page_tail
                bra send1

send2           ldx #page
                stx page_tail
                bra send1

send_end        puly
                pulx
                pulb
                pula
                rts

*************************************************
*  Prosedur DLY10 untuk delay selama 01 * 10 ms  *
*************************************************

dly10           pshx
                ldx #1
dly             pshx
                ldx #$0d06
dloop           dex
                bne dloop
                pulx
                dex
                bne dly

                pulx
                rts


*************************************************
* Prosedur KIRI untuk menggerakkan motor ke kiri *
*                  sebanyak STEP                *
*************************************************

kiri            psha
                pshx

                ldx step

kiri_1          ldaa portA
                anda #$04
                cmpa #$04
                bne kiri_3

                lsr stepX
                bcc kiri_2
```

```
            ldaa stepX
            adda #$80
            staa stepX

kiri_2      ldaa stepX
            anda #$0f
            staa portB

            jsr dly10

            dex
            bne kiri_1

kiri_3      ldaa #$0
            staa portB

            pulx
            pula
            rts

*************************************************
* Prosedur KANAN untuk menggerakkan motor ke kanan *
*                  sebanyak STEP                  *
*************************************************

kanan       psha
            pshx

            ldx step

kan_1       ldaa portA
            anda #$01
            cmpa #$01
            bne kan_3

            lsl stepX
            bcc kan_2
            ldaa stepX
            adda #$01
            staa stepX

kan_2       ldaa stepX
            anda #$0f
            staa portB

            jsr dly10

            dex
            bne kan_1

kan_3       ldaa #$0
            staa portB

            pulx
            pula
            rts
```

```
*************************************************
* Prosedur ATAS untuk menggerakkan motor ke atas *
*                  sebanyak STEP                  *
*************************************************

atas        psha
            pshx

            ldx step

atas_1      lsr stepY
            bcc atas_2
            ldaa stepY
            adda #$80
            staa stepY

atas_2      ldaa stepY
            anda #$f0
            staa portB

            jsr dly10

            dex
            bne atas_1

atas_3      ldaa #$0
            staa portB

            pulx
            pula
            rts

*************************************************
* Prosedur BAWAH untuk menggerakkan motor ke bawah *
*                  sebanyak STEP                  *
*************************************************

bawah       psha
            pshx

            ldx step

bwh_1       lsl stepY
            bcc bwh_2
            ldaa stepY
            adda #$01
            staa stepY

bwh_2       ldaa stepY
            anda #$f0
            staa portB

            jsr dly10

            dex
            bne bwh_1

bwh_3       ldaa #$0
            staa portB
```

```
            pulx
            pula
            rts

*****************************************
* prosedure ini untuk menyeleksi data *
*           dari noise                  *
* input  adalah BUFFER (2000 Byte)      *
* output adalah LPF    (2000 Byte)      *
*****************************************

filter      psha
            pshb
            pshx
            pshy

            ldx #buffer
            ldy #lpf
            ldaa #0
            staa 0,x
            staa 0,y

lpf_21      ldaa 1,x
            cmpa 0,x
            beq lpf_23

            cmpa 2,x
            beq lpf_23

            ldaa 0,x
            cmpa 2,x
            bne lpf_22

lpf_23      staa 1,y
            bra lpf_24

lpf_22      ldaa 1,x
            cmpa #0
            beq lpf_23
            ldaa 2,x
            staa 1,y

lpf_24      inx
            iny

            cpx #$47cf
            blt lpf_21

*-------------------------
*
*-------------------------
            ldx #10

lpf_10      ldy #lpf
            ldaa #0
            staa 0,y

            pshx
```

```
                ldx #2000

lpf_11          ldaa 1,y
                cmpa #0
                beq lpf_13

                ldaa 0,y
                cmpa #0
                beq lpf_12
                oraa 1,y
                staa 1,y

lpf_12          ldaa 2,y
                cmpa #0
                beq lpf_13
                oraa 1,y
                staa 1,y

lpf_13          iny
                dex
                bne lpf_11

                pulx
                dex
                bne lpf_10

                puly
                pulx
                pulb
                pula
                rts
******************
* prosedur TRANS
******************

trans           psha
                pshb
                pshx
                pshy

                ldx #lpf
                ldy #datanya

trans1          ldaa 0,x
                cmpa #0
                bne trans2
                inx
                cpx #$4faf
                blt trans1

                bra trans_end

trans2          ldaa 0,x
                staa 0,y
                iny

trans3          inx
                ldaa 0,x
                cmpa #0
```

```
            beq trans4
            cpx #$4faf
            bge trans_end
            bra trans3

trans4      pshy
            ldy #1

trans5      inx
            ldaa 0,x
            cmpa #0
            bne trans6
            iny
            cpx #$4faf
            blt trans5

            puly
            ldaa #0
            staa 0,y
            bra trans_end

trans6      cpy #10
            bge trans7
            puly
            bra trans2

trans7      cpy #20
            bge trans8
            puly
            ldaa #0
            staa 0,y
            iny
            bra trans2

trans8      cpy #30
            bge trans9
            puly
            ldaa #0
            staa 0,y
            iny
            staa 0,y
            iny
            bra trans2

trans9      puly
            ldaa #0
            staa 0,y
            iny
            staa 0,y
            iny
            staa 0,y
            iny
            bra trans2

trans_end   ldaa #0
            staa total
            ldy #datanya
            ldx #200
            ldaa #0
```

```
trans_tot  oraa Ø,y
           iny
           dex
           bne trans_tot

           staa total

           puly
           pulx
           pulb
           pula
           rts

*******************************
*  Prosedur untuk mengubah ke *
*       huruf braille         *
*******************************
to_braille psha
           pshb
           pshx
           pshy

           ldy #datanya
           ldx #200

to_brl1    ldaa #Ø
           staa simbol
           ldaa Ø,y
           anda #%00000110
           jsr cari
           oraa simbol
           staa simbol

           ldaa Ø,y
           anda #%00011000
           jsr cari
           oraa simbol
           staa simbol

           ldaa Ø,y
           anda #%01100000
           jsr cari
           oraa simbol
           staa simbol

           ldaa simbol
           staa Ø,y
           iny

           dex
           bne to_brl1

           puly
           pulx
           pulb
           pula
           rts
```

```
*************************
*      prosedur CARI
*************************

cari        pshb
            pshx
            pshy

            ldx #standar        ; offset STANDAR
            ldy #9

cari1       cmpa 0,x
            beq cari2
            inx
            dey
            bne cari1

            ldaa #0
            bra cari3

cari2       xgdx
            subd #standar
            addd #artinya
            xgdx
            ldaa 0,x

cari3       puly
            pulx
            pulb
            rts

*****************************************
*          prosedur TO_ASCII            *
* mengubah menjadi data dalam bentuk ASCII *
*     menyimpan ke PAGE pada posisi     *
*           PAGE_HEAD - 1               *
*****************************************
to_ascii    psha
            pshx
            pshy

            ldx #datanya
            ldy #transit
ascii1      ldaa 0,x
            cmpa #0
            bne ascii2
            inx
            cpx #$5150          ; end of DATANYA
            blt ascii1
            bra asciiend

ascii21     inx
            ldaa 0,x
            cmpa #0
            bne ascii22
            ldaa #%00010000
            bra ascii23
```

```
ascii2      ldaa 0,x
            staa ascii_reg
            cmpa #$01
            beq ascii21

            inx
            ldaa 0,x
ascii22     lsla
            lsla
            lsla
            lsla
            oraa ascii_reg
            inx

ascii23     staa 0,y
            iny
            cpy #$5a2a
            bge asciiend

ascii3      ldaa 0,x
            cmpa #0
            bne ascii2

            inx
            ldaa 0,x
            cmpa #0
            bne ascii2
            inx
            ldaa #0
            staa 0,y
            iny
            cpy #$5a2a
            blt ascii1

asciiend    ldaa #$00
            staa 0,y
            iny
            ldaa #$FF
            staa 0,y

            puly
            pulx
            pula
            rts

*********************************
* prosedur TO_PAGE              *
* untuk mengisi page dengan     *
* kode ascii yang akan dikirim *
*********************************
to_page     psha
            pshb
            pshx
            pshy

            ldx #transit
            dex
```

```
page1       ldaa full
            cmpa #$ff
            beq page1end

            inx
            ldaa 0,x
            cmpa #$ff
            beq page1end

            ldab #0             ; shift OFF
            stab shift

            cmpa #$10           ; shift ?
            bne page2

            ldaa #$80
            staa shift

            inx
            ldaa 0,x            ; ambil

            cmpa #$10           ; shift ?
            bne page2
            ldaa #$80
            staa shl
            ldaa #$00
            staa numlok
            bra page1

page1end    jmp page_end

page2       cmpa #$00
            bne page3
            ldaa #$00
            staa numlok
            staa shl
            ldaa #101
            staa ascii_reg
            jsr fill_page
            bra page1

page3       cmpa.#$71           ; numlock ?
            bne page4
            ldaa #$08
            staa numlok
            ldaa #$00
            staa shl
            bra page1

page4       cmpa #$33
            bne page5
            ldaa invers
            cmpa #$80
            bne page41
            ldaa #0
            staa invers
            ldaa #1

page40      staa ascii_reg
```

```
            jsr fill_page
            bra page1

page41      ldaa #$80
            staa invers
            ldaa #0
            bra page40

page5       oraa numlok
            oraa shl
            oraa shift

            jsr find
            cmpa #$ff
            beq page6
            staa ascii_reg
            jsr fill_page
            jmp page1

page6       ldaa 0,x
            jsr diftong
            cmpa #$ff
            beq page7
            staa ascii_reg
            jsr fill_page
            stab ascii_reg
            jsr fill_page
            jmp page1

page7       ldab 0,x
            inx
            ldaa 0,x
            jsr double
            cmpa #$ff
            beq page8
            staa ascii_reg
            jsr fill_page
            jmp page1

page8       ldaa #75
            staa ascii_reg
            jsr fill_page
            dex
            jmp page1

page_end    puly
            pulx
            pulb
            pula
            rts

************************
*  DOUBLE              *
************************
double      pshx
            pshy

            ldx #dblbrl
            ldy #4
```

```
double0     cpd 0,x
            beq double1
            inx
            inx
            dey
            bne double0

            ldaa #$ff
            bra doub_end

double1     xgdx
            subd #dblbrl
            addd #102
            tba

doub_end    puly
            pulx
            rts

************************
* DIFTONG              *
************************
diftong     pshx
            pshy

            ldx #dftbrl
            ldy #8
dift0       cmpa 0,x
            beq dift1
            inx
            dey
            bne dift0

            ldaa #$ff
            bra dift_end

dift1       xgdx
            subd #dftbrl
            addd #dftasc
            xgdx
            ldaa 0,x
            ldab 1,x

dift_end    puly
            pulx
            rts

************************
*       FIND           *
************************
find        pshb
            pshx
            pshy

            ldx #102
            ldy #braile

find1       cmpa 0,y
            bne find2
```

```
            xgdy
            subd #braile
            tba
            bra find_end

find2       iny
            dex
            bne find1

            ldaa #$ff
find_end    puly
            pulx
            pulb
            rts

*****************************
* prosedur FILL_PAGE        *
*                           *
* mengisi ascii_reg ke page *
*****************************
fill_page   pshy
            psha

            ldaa #0
            staa full

            ldy page_head
            cpy page_tail
            bne fill1
            ldaa #$ff
            staa full
            bra fill_end

fill1       ldaa ascii_reg
            dey
            staa 0,y

            ldy page_head
            cpy #$594f
            blt fill2

            ldy #page
            sty page_head
            bra fill_end

fill2       iny
            sty page_head

fill_end    pula
            puly
            rts

*********************************
* Prosedur ini adalah untuk
* menghapus semua memori
*********************************

clear_mem   psha
```

```
            pshx

            ldx #buffer

clear0      ldaa #0
            staa 0,x
            inx
            cpx #$5150
            blt clear0

            pulx
            pula
            rts

*****************************
*     inisialisasi SCI      *
*****************************
onsci       psha
            ldaa #$30
            staa baud
            ldaa #$00
            staa sccr1
            ldaa #$0c
            staa sccr2
            pula
            rts

*******************************
*     Kirim 1 Byte ke SCI     *
*                             *
*     kiriman di SCIBUF       *
*     error   di SCIWARN      *
*******************************

kirim       psha
            pshb
            pshx
            pshy

            ldx #$0fff

            ldy #scsr
kirim1      brset 0,y,#%10000000,kirim2
            dex
            bne kirim1

            ldaa #$ff
            staa sciwarn
            bra kirim3

kirim2      ldaa scibuf
            staa scdr
            ldaa #0
            staa sciwarn

            ldx #100
kirim21     jsr dly10
            dex
            bne kirim21
```

```
kirim3      puly
            pulx
            pulb
            pula
            rts

*******************************
*     Terima 1 Byte ke SCI    *
*                             *
*     output di SCIBUF        *
*******************************

terima      psha
            pshb
            pshx
            pshy

            ldy #scsr
terima1     brset 0,y,#%00100000,terima2
            jsr sw_press
            ldaa sw_flag
            cmpa #$ff
            bne terima1

            ldaa #$ff
            staa scibuf
            bra terima3

terima2     ldaa scdr
            staa scibuf

terima3     puly
            pulx
            pulb
            pula
            rts

*********************************
*       prosedur MORSE          *
*       inp = MORSEBUF          *
*********************************
morse       psha
            pshx
            pshy

            ldaa morsebuf
            cmpa #$80
            bne morse1
            ldy #50
            jmp morse7

morse1      ldx #8
morse2      dex
            lsl morsebuf
            bcs morse3
            cpx #0
            bne morse2
            jmp morse_end
```

```
morse3      dex
            lsl morsebuf
            bcs morse4
            ldy #75
            jsr bunyi
            bra morse5

morse4      ldy #225
            jsr bunyi

morse5      ldy #12
morse6      jsr dly10
            dey
            bne morse6

            cpx #0
            bne morse3

            ldy #25
morse7      jsr dly10
            dey
            bne morse7

morse_end   puly
            pulx
            pula
            rts

*****************************
* Prosedur bunyi
*****************************
bunyi       psha
            pshb
            pshx

            ldaa #$80
            staa oc1m

bunyi1      ldaa #$80
            staa oc1d

            ldd toc1
            addd #1666
            std toc1

bunyi2      ldx #tflg1
            brclr 0,x,#%10000000,bunyi2
            bclr 0,x,#%01111111

            ldaa #0
            staa oc1d

            ldd toc1
            addd #1666
            std toc1

bunyi3      ldx #tflg1
            brclr 0,x,#%10000000,bunyi3
```

```
hclr 0,x,#%01111111

dey
bne bunyi1

pulx
pulb
pula
rts

END
```

0 5 MAY 1993

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
ITS - SURABAYA

EE 1799 TUGAS AKHIR - 6 SKS

Nama Mahasiswa : NEMUEL DANIEL PAH
Nomor Pokok : 2882201112
Bidang Studi : ELEKTRONIKA
Tugas diberikan :
Tugas diselesaikan :
Dosen pembimbing : Ir. Nawantowibowo
Judul Tugas Akhir :

PEMBACA HURUF BRAILLE DENGAN MIKROKONTROLER 68HC11

Uraian Tugas Akhir :

Komunikasi merupakan kebutuhan penting dari semua umat manusia tak terkecuali bagi mereka yang mengalami cacat tubuh. Bagi penyandang tuna netra salah satu alat komunikasi adalah Huruf Braille yang tidak digunakan oleh yang tidak tuna netra. Untuk itu diperlukan alat yang dapat menerjemahkan naskah Braille ke naskah tertulis.

Pada Tugas Akhir ini akan dirancang dan dibuat alat yang dapat mengubah tulisan dalam naskah huruf Braille menjadi kode ASCII yang dikirim ke komputer sehingga naskah tersebut dapat dicetak, disimpan atau diolah lagi sesuai kebutuhan, serta juga diubah kedalam bentuk morse sehingga dapat didengar.

Alat ini diharapkan dapat membantu komunikasi antara kaum tuna netra dengan yang tidak tuna netra.

Surabaya, 29 April 1993

Menyetujui :
Bidang Studi Elektronika
Koordinator

Ir. Soetikno
NIP. 130 445 231

30/4/93

Dosen Pembimbing

Ir. Nawantowibowo
NIP. 130 368 612

Mengetahui
Jurusan Teknik Elektro
Ketua

Ir. Katjuk Astrowulan, MSEE
NIP. 130 687 483

USULAN TUGAS AKHIR

A. JUDUL TUGAS AKHIR : PEMBACA HURUF BRAILLE DENGAN MIKROKONTROLER 68HC11

B. BIDANG STUDI : ELEKTRONIKA

C. RUANG LINGKUP :
- Elektronika Mikro
- Pemrograman Komputer
- Elektronika Industri
- Mikroprosesor

D. LATAR BELAKANG : Komunikasi merupakan hal yang sangat penting bagi setiap manusia, tak terkecuali bagi mereka yang mengalami cacat tubuh. Bagi penyandang tuna netra, komunikasi tidak dapat dilakukan dengan indra penglihatan yaitu mata melainkan dengan indra lainnya yang masih aktif. Untuk membaca digunakan naskah dengan huruf Braille sehingga mereka dapat menggunakan indra kulit, dalam hal ini jari untuk membaca.

Bagi yang memiliki indra yang lengkap naskah ditulis dengan tinta sehingga jarang sekali dapat membaca naskah dalam huruf Braille, demikian juga sebaliknya. Hal ini menuntut adanya alat yang dapat menerjemahkan naskah dari kedua bentuk tersebut sehingga baik yang tuna netra maupun tidak, dapat saling berkomunikasi dengan lancar, khususnya dalam bentuk tertulis.

E. PENELAAHAN STUDI : Huruf Braille adalah huruf yang digunakan oleh kaum tuna netra sebagai alat komunikasi tertulis. Huruf ini berupa kombinasi dari 6 (enam) buah relief titik yang letaknya beraturan. Huruf ini dibaca oleh kaum tuna netra dengan cara menyentuhkan jari pada relief tersebut.

Peralatan yang akan dibuat terdiri dari tranduser yang akan mengubah besaran relief dari huruf Braille menjadi besaran listrik. Besaran ini kemudian disusun dalam bentuk 8 bit (byte) yang kemudian diolah dengan perangkat lunak untuk diperoleh pola standar sehingga besaran listrik yang dikirim oleh tranduser tersebut dapat dimengerti. Data-data tersebut diubah ke dalam kode ASCII dan disimpan dalam memory. Untuk menampilkan data tersebut pada komputer maka data tersebut akan dikomunikasikan dengan komputer. Data yang ada di komputer berupa teks huruf latin akan dapat dicetak atau disimpan ke media penyimpan. Selain dikirim ke komputer maka data tersebut juga diubah ke dalam bentuk morse sehingga dapat didengar oleh penyandang tuna netra.

Untuk mengolah data, menyimpan data mengatur gerakan dari motor, dan semua kegiatan dari peralatan ini, digunakan mikrokontroler 68HC11.

F. TUJUAN : Dalam Tugas Akhir ini akan dirancang dan dibuat peralatan yang dapat membaca naskah huruf Braille dan diubah menjadi naskah huruf latin.

G. LANGKAH-LANGKAH :
- Studi literatur dan perancangan peralatan
- Pembuatan peralatan
- Pengujian peralatan
- penulisan naskah Tugas Akhir

H. JADWAL KEGIATAN : Pembuatan Tugas Akhir ini akan dilakukan selama enam bulan dengan jadwal sebagai berikut:

KEGIATAN	BULAN KE-					
	I	II	III	IV	V	VI
1. Studi literatur & perancangan peralatan	■	■	■			
2. Pembuatan peralatan		■	■	■	■	
3. Pengujian peralatan				■	■	
4. Penulisan naskah Tugas Akhir					■	■

I. RELEVANSI : Dengan peralatan ini naskah huruf Braille dapat dicetak sebagai naskah latin, disimpan dalam media penyimpan, serta jika dirangkaikan dengan printer huruf Braille maka naskah Braille dapat diperbanyak.

RIWAYAT HIDUP

Nemuel Daniel Pah, dilahirkan di Kupang NTT pada tanggal 31 Desember 1969, sebagai putra kedua dari tiga bersaudara, dari Bapak Daniel N Pah dan Ibu Strina Ndoen, yang bertempat tinggal di jalan WJ. Lalamentik 55 Kupang NTT.

Terdaftar sebaghai mahasiswa Institut Teknologi Sepuluh Nopember Surabaya, Fakultas Teknologi Industri, Jurusan Teknik Elektro pada tahun 1988 dengan nomor pokok 2882201112. Selama menjadi mahasiswa pada tahap sarjana aktif menjadi asisten praktikum Rangkaian Listrik, Elektronika, dan Elektronika Lanjut II.

Pendidikan yang ditempuh sampai saat ini :

- SD Negeri X, Naikoten I, Kupang.
- SMP Negeri 2 Kupang.
- SMA Negeri 1 Kupang.
- Perguruan Tinggi di Jurusan Teknik Elektro, FTI-ITS, dan diharapkan menyelesaikan studinya pada Ujian Sarjana Periode Maret 1994.